Decentralized multi-robot cooperation with auctioned POMDPs

Jesus Capitan, Matthijs T.J. Spaan, Luis Merino and Anibal Ollero

The International Journal of Robotics Research 2013 32: 650
DOI: 10.1177/0278364913483345

The International Journal of
Robotics Research
32(6) 650–671

DOI: 10.1177/0278364913483345
ijr.sagepub.com

Decentralized multi-robot cooperation with auctioned POMDPs

Jesus Capitan[1], Matthijs T.J. Spaan[2], Luis Merino[3] and Anibal Ollero[4]

Abstract

Planning under uncertainty faces a scalability problem when considering multi-robot teams, as the information space scales exponentially with the number of robots. To address this issue, this paper proposes to decentralize multi-robot partially observable Markov decision processes (POMDPs) while maintaining cooperation between robots by using POMDP policy auctions. Auctions provide a flexible way of coordinating individual policies modeled by POMDPs and have low communication requirements. In addition, communication models in the multi-agent POMDP literature severely mismatch with real inter-robot communication. We address this issue by exploiting a decentralized data fusion method in order to efficiently maintain a joint belief state among the robots. The paper presents two different applications: environmental monitoring with unmanned aerial vehicles (UAVs); and cooperative tracking, in which several robots have to jointly track a moving target of interest. The first one is used as a proof of concept and illustrates the proposed ideas through different simulations. The second one adds real multi-robot experiments, showcasing the flexibility and robust coordination that our techniques can provide.

Keywords

multi-robot cooperation, planning under uncertainty, decentralized data fusion

1. Introduction

Multi-robot systems are of great interest in many robotic applications, such as exploration (de Hoog et al., 2009), surveillance (Burdakov et al., 2010), monitoring (Leonard et al., 2010) or rescue robotics (Merino et al., 2006; Hsieh et al., 2007; Maza et al., 2011). In those applications, a single robot is not usually able to acquire all of the required information and the cooperation among multiple robots is essential (see Figure 1). However, real scenarios present uncertain and potentially hazardous environments in which robots can experience communication constraints regarding connectivity, bandwidth and delays. Mapping the overall task into robust plans for each robot is a challenging problem.

Even when planning for a single robot, uncertainty complicates task planning. For instance, in many robotic applications the sensors on board the robot do not allow it to unambiguously identify its own location or pose (Thrun et al., 2005). Partially observable Markov decision processes (POMDPs) provide a sound mathematical framework to cope with decision-making in uncertain and partially observable environments (Sondik, 1971; Kaelbling et al., 1998).

However, although there are POMDP solvers able to successfully handle large state spaces currently, POMDPs ultimately face a scalability problem when considering planning for multi-agent teams (Seuken and Zilberstein, 2008). Popular models such as Dec-POMDPs (Bernstein et al., 2002) or ND-POMDPs (Nair et al., 2005) remain limited to toy problems, and other models require flawless instantaneous communication (Pynadath and Tambe, 2002; Nair et al., 2004; Roth et al., 2005).

In this paper, we propose a scheme for exploiting the power of decision-theoretic planning methods such as POMDPs, while mitigating their complexity by lowering the dependence between individual plans. In particular, the approach solves independent POMDPs for each robot, but still fosters online cooperation during the execution phase

[1]University of Duisburg-Essen, Duisburg, Germany
[2]Delft University of Technology, Delft, The Netherlands
[3]Pablo de Olavide University, Seville, Spain
[4]University of Seville, Seville, Spain

Corresponding author:
Jesus Capitan, University of Duisburg-Essen, Bismarckstr. 90, 47057, Duisburg, Germany.
Email: jescap@cartuja.us.es

Fig. 1. Multi-robot systems such as multi-UAV systems have been demonstrated as very useful in tasks such as disaster monitoring, tracking or surveillance activities (Merino et al., 2006; Maza et al., 2011).

by distributing the individual policies using auctions. Auction algorithms have been widely used for optimal multi-robot task allocation (Gerkey and Matarić, 2004; Mosteo and Montano, 2007; Viguria et al., 2008), and have also been explored in conjunction with POMDPs (Spaan et al., 2010a).

In addition, robotic teams commonly are capable of communicating, which we exploit to maintain a decentralized state estimate. Nonetheless, a key point in our approach is that we relax the strict assumptions on the quality of the communication channel commonly found in the literature on multi-agent planning under uncertainty (Pynadath and Tambe, 2002; Nair et al., 2004; Roth et al., 2005). Actually, we consider fully decentralized solutions, that is, solutions that only involve local information and point-to-point communications, and which are scalable with the total number of robots.

In the following subsections, a definition of the decentralized models assumed in this paper is stated; some related works are reviewed; and the main contributions of the paper are highlighted.

1.1. Decentralized systems

When designing control algorithms for multi-robot systems, it is important to keep in mind their possibilities and

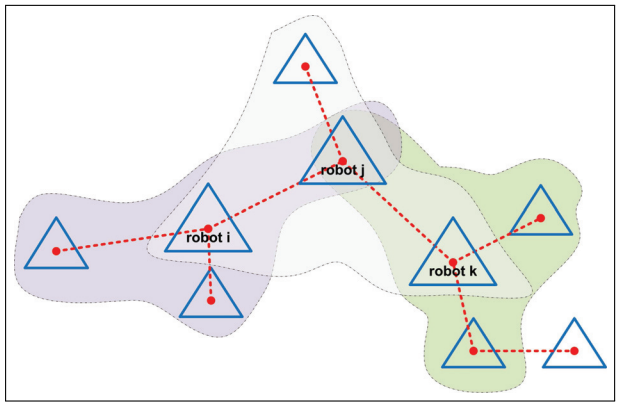

Fig. 2. Decentralized approach. The communication domains for robots i, j and k at a given instant are indicated. All of the operations are performed within these domains, so the complexity of the operations is limited by the number of neighbors of each robot. However, the information flow through the network will permit cooperation between all robots.

constraints. Of particular relevance in our context is the fact that communication between robots is often possible, but the quality of the communication channel can vary. This precludes centralized solutions as well as methods requiring communication guarantees. Therefore, we focus on decentralized models. In particular, we adhere to the following definition of a decentralized system (Nettleton et al., 2003), see Figure 2:

1. There is no central entity required for the operation.
2. There is no common communication facility; that is, information cannot be broadcasted to the whole team, and only local point-to-point communications between neighbors are considered.
3. The robots do not have a global knowledge about the team topology: they only know about their local neighbors.

These characteristics make the system scalable as it does not require a central node and enough bandwidth to transmit all of the information to that node. Moreover, the system is more robust and flexible with respect to loss or inclusion of new robots (there is no need to know the global topology), and with respect to communication issues (a failure does not compromise the whole system).

1.2. Related work

In the single-robot case, the POMDP model has been applied to a wide variety of robotic applications. Those include robot navigation (Simmons and Koenig, 1995; Spaan and Vlassis, 2004; Roy et al., 2005; Foka and Trahanias, 2007), active sensing (Hoey and Little, 2007; Spaan et al., 2010b), object grasping (Hsiao et al., 2007) or human–robot interaction (Doshi and Roy, 2008).

For multi-robot systems, there are also many models extending POMDPs, but they usually make strong assumptions in terms of communication quality or face scalability issues regarding the number of robots. This literature for multi-agent planning under uncertainty will be reviewed later in Section 3.

Apart from POMDPs, there are many other probabilistic approaches that deal with decision-making in multi-robot teams. For instance, a multi-task architecture for tracking and monitoring is presented by Maza et al. (2011). However, reasoning on the uncertainties inherent to the different tasks is just included as an *ad hoc* plan refining module, as the tasks are considered observable and deterministic. Moreover, a decentralized algorithm for mapping and tracking is presented by Stroupe and Balch (2005), whereas Bourgault and Durrant-Whyte (2004) and Wong et al. (2005) focus on multi-target search. Also, there is some work (Shah et al., 2009) dealing with online task allocation under uncertainties in which the robots reason about temporal constraints. In a close work to ours but using a different model, Cole et al. (2006) consider decentralized state estimation and auctioning for multi-vehicle coordination in information gathering missions. Optimal policies are obtained by maximizing the expected information gain as utility. Multi-vehicle coordination is performed by a similar auctioning mechanism. Even though these approaches may cope with some of them, POMDP techniques embrace together several interesting features for robotic tasks: (i) they deal with sequential problems and optimize forward for a time horizon; (ii) they reason about noisy and delayed robot movements; (iii) they can weight naturally different cost functions in parallel and implicitly consider the value of information gathering.

1.3. Main contributions

As a first contribution in this paper, we propose to emulate a multi-robot POMDP by combining individual behaviors or roles that can be represented by single-robot POMDPs. We generalize a centralized POMDP auction (Spaan et al., 2010a) to assign never-ending policies (behaviors) to different robots at every step. In this novel decentralized auction, instead of tasks, POMDP policies that describe a behavior towards a common goal are distributed; robots can switch between these behaviors dynamically at each decision step. The auction determines continuously which behavior is best for each robot to cooperatively attain the goal. Since local POMDPs are solved for each robot, the inter-connection between the models is low and the approach can scale well with the number of robots.

The second key component is to efficiently maintain a joint belief state among the robots, which can serve as coordination signal. We use an existing decentralized data fusion (DDF) approach (Capitan et al., 2011), but in conjunction with POMDP policies for a multi-robot system. Unlike most work on POMDPs, the belief update here is separated from the decision-making process during the execution phase. This decoupling between both processes increases the robustness and reliability of real-time robotic teams.

In order to highlight the flexibility and performance of the methods proposed, two different multi-robot applications are presented. First, environmental monitoring, in which several unmanned aerial vehicles (UAVs) have to evaluate the level of contamination on a given terrain with less uncertainty as possible. Second, a tracking application, in which several robots have to cooperate in order to track a moving target as accurately as possible. Nonetheless, our techniques suit a wider range of problems, such as surveillance (Hsieh et al., 2007; Burdakov et al., 2010) or fire detection (Merino et al., 2006; Maza et al., 2011), which call for a cooperative effort of robots coordinating their individual roles.

The paper shows extensive simulation results, but it also demonstrates the proposed approaches in a real multi-robot testbed. This is done for the cooperative tracking application and using a fully decentralized setup in real-time.

1.4. Organization

The remainder of the paper is organized as follows: Section 2 summarizes POMDP models and describes the DDF algorithms; Section 3 discusses current approaches in the literature for multi-agent planning under uncertainty; Section 4 proposed a role-based model for multi-robot planning; Section 5 describes the algorithms for auctioning POMDPs in a decentralized manner and the overall overview of the complete system; Sections 6 and 7 present the two applications used as case studies, including experimental results; and Section 8 gives the conclusions and future work.

2. Background

We give a short description of a decision-theoretic model for single-robot and multi-robot planning, followed by a method for maintaining a joint belief by multiple robots.

2.1. Decision-theoretic planning models

A popular model for single-robot planning under uncertainty in sensing and acting is the POMDP.

Formally, a POMDP is defined by the tuple $\langle S, A, Z, T, O, R, h, \gamma \rangle$ (Kaelbling et al., 1998): the *state space* is the finite set of possible states $s \in S$; the *action space*, the finite set of possible actions $a \in A$; and the *observation space* consists of the finite set of possible observations $z \in Z$. At every step, an action is taken, an observation is made and a reward is given. Thus, after performing an action a, the state transition is modeled by the conditional probability function $T(s', a, s) = p(s'|a, s)$, and the posterior observation by the conditional probability function $O(z, a, s') = p(z|a, s')$. The reward obtained at each step is $R(s, a)$, and the objective is to maximize the sum of expected rewards, or *value*, earned during h time

steps. To ensure that this sum is finite when $h \to \infty$, rewards are weighted by a discount factor $\gamma \in [0, 1)$.

Given that it is not directly observable, the actual state cannot be known by the system. Instead, a probability density function $b(s)$ over the state space is maintained. This is called the *belief state* and, due to the Markov assumption, it can be updated with a Bayesian filter for every action–observation pair:

$$b'(s') = \eta O(z, a, s') \sum_{s \in S} T(s', a, s)\, b(s) \tag{1}$$

where η acts as a normalizing constant such that b' remains a probability distribution.

The objective of a POMDP is to find a policy that maps beliefs into actions in the form $\pi(b) \to a$, so that the total expected reward is maximized. This expected reward gathered by following π starting from belief b is called the value function:

$$V(b) = E\left[\sum_{t=0}^{h} \gamma^t r(b_t, \pi(b_t)) \,|b_0 = b\right] \tag{2}$$

where $r(b_t, \pi(b_t)) = \sum_{s \in S} R(s, \pi(b_t))\, b_t(s)$. Therefore, the optimal policy π^* is the one that maximizes that value function: $\pi^*(b) = \arg\max_{\pi} V(b)$.

There are two key results for computing optimal policies in discrete POMDPs. First, the value function at horizon t can be constructed iteratively from the value function at horizon $t-1$ (value iteration):

$$V^{(t)}(b) = \max_a \left[\sum_{s \in S} R(s, a)\, b(s) + \gamma \sum_{z \in Z} p(z|a, b)\, V^{(t-1)}(b_a^z)\right] \tag{3}$$

where b_a^z is the belief updated according to (1). Second, it can be proven that the optimal value function for any finite horizon is piecewise linear and convex (Sondik, 1971). It means that the value function at iteration t can be expressed by a set of vectors Γ_t ($|S|$-dimensional hyperplanes), each of them defining a region in the belief space for which they are the maximizing element of $V^{(t)}$.

In general, computing an optimal policy requires one to explore the continuous belief space, which can be very complex. Classic point-based solvers, such as SARSOP (Kurniawati et al., 2008) or Symbolic Perseus (Poupart, 2005), approximate the value function iteratively (3) using a finite set of belief points. Moreover, other solvers (Theocharous and Kaelbling, 2003; He et al., 2011; Kurniawati et al., 2011) improve policy computation by sampling the belief space efficiently or by considering macro-actions. However, our approach is independent of the particular (approximate) POMDP solver employed, and advances in POMDP solving can be applied directly.

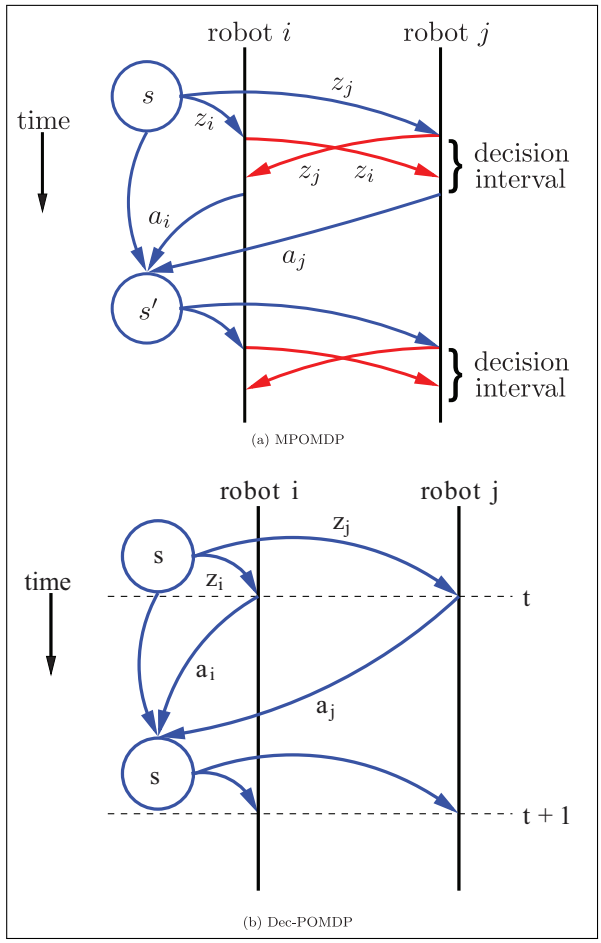

Fig. 3. Illustration of the difference in information flow between a centralized model (a) and a decentralized one (b).

2.2. *Multi-robot models*

When a set of n robots that share the same reward function is considered, there are two main options to extend the POMDP framework. The Decentralized POMDP (Dec-POMDP) model allows for fully decentralized execution (Bernstein et al., 2002), while the multi-agent POMDP (MPOMDP) takes a centralized approach (Pynadath and Tambe, 2002).

Each robot i can execute an action a_i from a finite set A_i and receives an observation z_i from a finite set Z_i. The key difference between the Dec-POMDP and MPOMDP models is that in the decentralized case, each robot only observes its local observation z_i, while in the centralized case, each robot observes the full observation vector z. Figure 3 illustrates these concepts, showing that in an MPOMDP the robots need to synchronize their knowledge using communication. In fact, for the reduction to an MPOMDP, communication is assumed to be instantaneous and free of cost (Pynadath and Tambe, 2002).

In both models, the transition function $T(s', a_J, s)$ is defined over the set of joint actions $a_J \in A_1 \times \cdots \times A_n$, and the observation function $O(z_J, a_J, s')$ relates the state to

the joint action and the joint observation $z_J \in Z_1 \times \cdots \times Z_n$. The common reward signal is defined over the joint set of states and actions $R : S \times A_1 \times \cdots \times A_n \rightarrow \mathbb{R}$.

The goal in the multi-robot case is to compute an optimal joint policy $\pi^* = \{\pi_1, \ldots, \pi_n\}$ that maximizes the expected discounted reward (as in the POMDP case). In the MPOMDP case, as robots at each time step have access to the joint observation and the joint action, they can maintain a joint belief using (1) (substituting the single-robot models with the joint ones). In the Dec-POMDP case, beliefs over the state space cannot be computed as the observation function requires the full observation vector to be known. As a result, optimal policies map individual action–observation histories to actions. The computational complexity of solving a Dec-POMDP is significantly higher than that of a POMDP (NEXP-complete (Bernstein et al., 2002) versus PSPACE-complete (Papadimitriou and Tsitsiklis, 1987)).

2.3. Decentralized data fusion

Between the information exchange required by the MPOMDP model (equivalent to a centralized system with access to all of the information at every time instant) and the Dec-POMDP model (no exchange at all), there is the option of estimating the joint belief (1) in a decentralized way: that is, using just local information and exchanging information with the neighbors.

In order to determine how to estimate the belief in a decentralized way, it is convenient to consider the joint belief $b_J(s_{0:t})$ for the full state trajectory (from time 0 up to time t). Assuming that the data gathered by the different robots at any time instant t are *conditionally independent* given the state at that instant s_t, and the same assumptions as in (1), the Bayes filter to compute this joint belief state is given by

$$b_J(s_{0:t}) = \eta' \prod_{\tau=1}^{\tau=t} \Big[\prod_{i=1}^{n} O(z_{i,\tau}, a_{J,\tau}, s_\tau) \Big] T(s_\tau, a_{J,\tau}, s_{\tau-1})\, b(s_0) \quad (4)$$

with $b(s_0)$ the prior and η' a normalization constant (we assume, without loss of generality, that every robot gathers one measurement at every time instant). This filter requires access to all of the information provided by the team at any moment.

In a decentralized approach, however, each robot employs only its local data z_i and then *shares* its belief with its neighbors at certain time instants. The received information from other teammates is locally fused in order to improve the local perception of the world. The local belief state over the full trajectory $b_i(s_{0:t})$ for robot i is

$$b_i(s_{0:t}) = \eta'' \prod_{\tau=1}^{\tau=t} O(z_{i,\tau}, a_{i,\tau}, s_\tau)\, T(s_\tau, a_{J,\tau}, s_{\tau-1})\, b(s_0)\,. \quad (5)$$

Comparing this expression to (4), it is possible to obtain the centralized joint belief from the local ones:

$$b_J(s_{0:t}) = \eta \prod_{i=1}^{n} \frac{b_i(s_{0:t})}{b(s_{0:t})_0} b(s_{0:t})_0 \quad (6)$$

where $b(s_{0:t})_0 = \prod_{\tau=1}^{\tau=t} T(s_\tau, a_{J,\tau}, s_{\tau-1})\, b(s_0)$.

Then, if a robot of the team receives all of the beliefs from the other robots, the fusion operation consists of combining all of the local beliefs after removing the common information they share (the prior over the trajectory $b(s_{0:t})_0$). The important aspect to be pointed out is that, by applying this equation, the centralized belief can be *exactly* recovered.

Several aspects should be considered. First, in a decentralized system one robot will communicate just with some neighbors. Therefore, (6) will result on a belief different from the joint one, but which summarizes all of the information from those neighbors. Later, when this belief is communicated throughout the network, the joint belief will be recovered eventually. For example, in Figure 2, robot k will have eventually a common shared belief with the rest of the team thanks to its communication with robot j; the belief obtained by robot j summarizes all of the information obtained by its neighbors, including those not directly in contact with robot k, such as robot i (which itself summarizes all of the information from its neighbors).

Second, the communication between robots will happen from time to time. Every time fusion operations are performed shared beliefs are recovered, but meanwhile there may be a desynchronization between local beliefs. Moreover, applying (6) requires to know the actions of the other robots (to apply $T(s_\tau, a_{J,\tau}, s_{\tau-1})$). These actions could be communicated, or even predicted from the fused beliefs (in case no communication is available).

Finally, not only does each robot receive information from its neighbors, but also sends information to them. In this case, the fusion equation is slightly different. If robot i received information from j, its belief would be updated as follows:

$$b_i(s_{0:t}) \leftarrow \eta \frac{b_i(s_{0:t})\, b_j(s_{0:t})}{b_{ij}(s_{0:t})} \quad (7)$$

where $b_{ij}(s_{0:t})$ represents the common information between the robots (i.e. the common prior mentioned above $b(s_{0:t})_0$, but also information previously exchanged between the robots). This common information can be maintained by a separate filter called channel filter (Bourgault and Durrant-Whyte, 2004). If there are loops in the information channels, the problem of double counting should be taken into account as well.

As commented, the previous equations are exact even for dynamic states. However, maintaining a belief for the state trajectory is very costly. Capitan et al. (2011) have presented an algorithm for DDF that scales only linearly with the length of the trajectory, under the assumption of Gaussian beliefs. For other belief functions, the same Equation (7) can be applied to the belief on the last state $b(s_t)$.

However, some error will be committed with respect to the centralized ideal belief if the fusion equation is not applied every time instant in which a measurement is obtained in the team. This error will depend on the effect of the prediction function on the belief. In the work of Bourgault and Durrant-Whyte (2004) it can be seen how this error stays bounded for a searching application in which the dynamics of the target implies a zero-mean Gaussian diffusion of the belief, and its size depends on the communication rate of the DDF.

Furthermore, Capitan et al. (2011) and Merino et al. (2012) show the error committed in similar tracking applications. They highlight that the error between communication instants also depends on the amount of information associated with the measurements gathered by the fleet in the meantime (represented by the information matrix of these measurements in the Gaussian case). In that case, as a delayed-state filter is considered, the error is reset in each communication event.

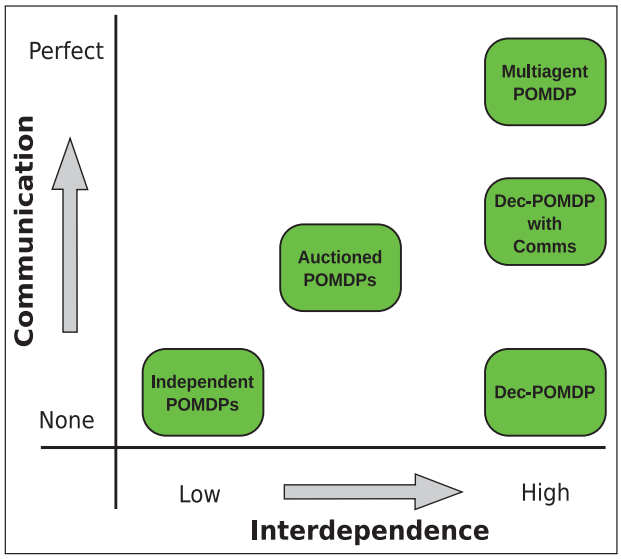

Fig. 4. Classification of MPOMDP approaches according to interdependence and level of communication between the agents. "Auctioned POMDPs" refers to our proposed approach.

3. Multi-agent planning under uncertainty

In the literature a wide variety of decision-theoretic models exist to deal with multi-agent systems (Seuken and Zilberstein, 2008), of which MPOMDPs and Dec-POMDPs were already introduced in Section 2.2. However, many of these models have severe drawbacks when considering applying them to multi-robot scenarios. Before presenting our solution, we analyze the models available in the literature by comparing them in terms of agent interdependence and communication assumptions.

The level of interdependence between agents is determined by 1) the amount of information that an agent needs to know about the other agent and 2) how coupled the final policies are. We call a system highly interdependent if a change in one of the agents' model requires re-computing the policies for the others. Many models from the literature are highly interdependent, for instance multi-agent Markov decision processes (MMDP) (Boutilier, 1996), MPOMDPs, Dec-POMDPs, and ND-POMDPs (Nair et al., 2005), and I-POMDPs (Gmytrasiewicz and Doshi, 2005). Figure 4 presents a possible classification of existing models with respect to their interdependence and the grade of communication that is assumed for the agents.

The simplest approach is to map the global task as well as possible into a set of individual tasks, and model these as independent POMDPs (Figure 4, bottom left). Thus, each agent can solve its own POMDP and execute its own policy without any communication. In this case, the interdependence between agents is very low, but since each agent ignores the others, the level of cooperation or even coordination is low too. Many interesting multi-agent planning problems cannot be tackled adequately with such a loosely coupled approach. The advantage of such an approach is its relatively low computational complexity, since it only requires solving n single-agent POMDPs, each of which is defined over individual action and observation spaces. Hence, scalability in the number of agents is linear, which is very low compared with other models.

On the other end of the spectrum, MPOMDPs and Dec-POMDPs solve a single decision-theoretic model for the whole team reasoning about all of the actions and observations of each agent (Figure 4, right column).

The MPOMDP model assumes perfect communication and each agent has access to joint actions and observations at every moment, whereas the Dec-POMDP model assumes no communication at all. Such models allow for tight coordination, but they present a high interdependence, since any small change in one of the agents entails a recalculation of the policy for the whole team. Furthermore, if due to imperfect communication agents do not have access to other agents' observations, the behavior of the MPOMDP model is not defined. Regarding computational complexity, an MPOMDP is a POMDP defined over the joint action and observation spaces, whose sizes grow exponentially with the number of agents.

The Dec-POMDP model, on the other hand, does not exploit communication at all, which in many scenarios could be beneficial to improve team performance. For instance, in our cooperative tracking application, it is easy to see that if a pursuer robot detects the target and informs its teammates about the target's approximate location, the other pursuers can close in on the target. Without communication, each pursuer might need to find the target by itself, which is clearly less time-efficient. Solving Dec-POMDPs optimally takes doubly-exponential time in the worst case, which severely restricts their applicability in multi-robot scenarios.

In between MPOMDPs and Dec-POMDPs there are several models in which some communication is

assumed (Nair et al., 2004; Roth et al., 2005; Spaan et al., 2008; Oliehoek and Spaan, 2012) (Figure 4, middle right). These models try to exploit the fact that agents actually share information, but just partially and at certain instants. Furthermore, most of them assume that communication arrives instantly.

By looking at the current state of the literature, we can conclude existing multi-agent decision-theoretic models do not take into account the requirements that multi-robot missions pose. First, a critical dependence on communication is to be avoided, but it should be exploited when available. Second, a strong coupling between individual robots is undesirable, as tightly coordinated joint actions are often hard to execute with a low probability of success.

4. Role-based multi-robot POMDP

In order to address the shortcoming of existing multi-agent planning models for multi-robot scenarios, as discussed in Section 3, we present a new model that specifically takes into account multi-robot issues. In a sense, we aim to reach middle ground on both axes of Figure 4.

There are many multi-robot missions that could be modeled as POMDPs (Kok et al., 2005; de Hoog et al., 2009; Burdakov et al., 2010). If all the robots have access to joint information (actions and observations from the whole team), the problem can be modeled as a MPOMDP. The objective of the team can be encoded in a reward function that, in general, depends on joint states and joint actions, and can be seen as the addition of the local rewards for each robot:

$$R(s,a) = R_1(s,a) + \cdots + R_n(s,a). \tag{8}$$

Without losing generality, the reward can be decomposed into two parts: one based only on local information R_i^{local} from each robot i; and one based on joint information R^{joint}. The local information for a robot i means its action a_i and its state s_i.[1] In case of a factored state, each local state s_i would include the local factors that can be controlled by local actions, and the factors that are common for all of the robots. Thus, the global reward can be expressed as

$$R(s,a) = R_1^{local}(s_1,a_1) + \cdots + R_n^{local}(s_n,a_n) + R^{joint}(s,a). \tag{9}$$

Apart from the *local* rewards (i.e. the rewards that robots would get if there were no others), there is the coupled term $R^{joint}(s,a)$, which models cooperation among the robots. Indeed, actions from different robots need to be considered in order to compute this reward. Even though the design of this cooperative term is very dependent on the application, in many cases, due to efficiency issues, it is common to penalize different robots repeating similar tasks. For instance, in many surveillance applications the robots should get less reward for surveying an area that is already being surveyed by another.

The previous idea is useful for many robotic applications in which there are either limited resources that cannot be accessed simultaneously by the robots, or different roles/tasks that must be covered. Thus, the team objective in many missions (e.g. detecting a target or alarm) can be achieved with robots following different roles or behaviors (e.g. patrol a certain area, approach the target, etc.). For instance, in smart energy grids there are providers and consumers (van der Sluis, 2011); and in robotic soccer strikers and defenders (Kok et al., 2005). Also, in active perception applications (Maza et al., 2011), where the team needs to maximize its information, it is positive to have robots following non-overlapping behaviors in order to provide richer information to the team.

In this work, we are interested in these role-based applications. In order to model the problem that way, we assume the following.

Assumption 1. *For a given mission, there is a finite set of m possible roles/behaviors for the robots that are exclusive and define the whole problem. In other words, the problem can be modeled with a set of m non-overlapping behaviors (i.e. each robot can only be playing one role at a moment).*

Assumption 2. *Each role is a single-robot behavior that can be represented by a reward function $R_i^k(s_i,a_i)$, where $k \in \{1,\ldots,m\}$ identifies the behavior and $i \in \{1,\ldots,n\}$ the robot performing it.*

Provided that $\{c_i\}_{i=1,\ldots,n} \in \{1,\ldots,m\}$ are the behaviors chosen by each robot, the local rewards in (9) depend on this assignment. These rewards R_i^{local} are those that each robot would get by acting on its own, and they are encoded in the corresponding $R_i^{c_i}$. Moreover, the cooperative term also depends on the assignment of the behaviors: $R^{joint} = f(s,a,c_1,\ldots,c_n)$.

The idea in our role-based model is that, at each time step, the robots should select their behaviors optimally (apart from their actions) in order to maximize the expected reward of the whole team. Therefore, we can define a role-based MPOMDP in which the value function (3) is modified as follows:

$$V^{(t)}(b) = \max_{a,c_{1,\ldots,n}} \{\sum_{s\in S} [R_1^{c_1}(s_1,a_1) + \cdots + R_n^{c_n}(s_n,a_n) + R^{joint}(s,a,c_{1,\ldots,n})]b(s) + \gamma \sum_{z\in Z} p(z|a,b)\, V^{(t-1)}(b_a^z)\}. \tag{10}$$

This new value iteration performs optimization at two different levels, behavior level and action level. Actions and behaviors are chosen at each step so that the value function is maximized. Note that the role assignments from one step to the next one are not correlated. This role-based model may entail even higher computational complexity than the original MPOMDP, since the optimization is carried out over all of the possible joint actions and behavior assignments. In the next section, we propose an approximate

method to solve the role-based MPOMDP in which the policies are sub-optimal, but the computational complexity of the solution is reduced dramatically.

5. Decentralized auction with POMDPs

The proposed approach builds on two mechanisms: the DDF filter described in Section 2.3 and a POMDP auction (Section 5.1). The former allows the robots to share information and build a joint belief like in a MPOMDP, the latter is used to assign the different behaviors to the robots in a cooperative manner. Moreover, both approaches are decentralized following the definition of Section 1.1.

In Figure 4, our approach can be seen as in between "independent POMDPs" and MPOMDP/Dec-POMDP in terms of agent interdependence. In terms of communication requirements, our approach does not require the high-quality guarantees of the methods that enhance the Dec-POMDP model with communications.

5.1. Auctioning POMDP policies

We already stated in Section 4 how a MPOMDP can be modeled with single-robot behaviors for a certain class of problems. Such a decomposition is of interest in many robotic applications such as search and rescue (Hsieh et al., 2007; Burdakov et al., 2010), tracking (Maza et al., 2011), fire detection (Merino et al., 2006) or robotic soccer (Kok et al., 2005), in which cooperation between robots playing different roles is required. An idea to solve the role-based MPOMDP is to use the reward functions of the behaviors to define a set of single-robot POMDPs that can be solved separately offline. Then, these behaviors can be run online simultaneously and combined in some optimal manner to produce a joint behavior similar to that desired for the whole team initially.

The objective is to approximate the multi-robot reward in (9). For that, a single-robot POMDP is defined for each given behavior $k \in \{1,\dots,m\}$ and robot $i \in \{1,\dots,n\}$. With each POMDP the reward function of the corresponding behavior R_i^k is associated, which is defined over the sets of local variables $\langle S_i, A_i, Z_i \rangle$. In an offline planning phase, individual policies are computed for all of these POMDPs, each of them with a value function $V_i^k(b_i)$. Although the actual multi-robot objective cannot be modeled as a set of single-robot reward functions, if these policies could be assigned optimally to one or more robots, all together should lead to a cooperative behavior pursuing the global objective. The problem of determining which policy should be assigned to each robot at each step can be modeled as a task allocation problem (Spaan et al., 2010a).

In general, a task allocation algorithm attempts to assign a set of m tasks to a team of n robots minimizing a global cost. In this case, each robot always has to be assigned a sole task, which is the POMDP policy to follow. In order to foster cooperation, we try not to assign the same policy to different robots. Given that $x_{ik} = 1$ when policy k is assigned to robot i ($c_i = k$) and 0 otherwise, and ϕ_{ik} is the cost associated with that assignment, the problem consists of minimizing the total cost:

$$\min \sum_{i=1}^{n} \left(\sum_{k=1}^{m} \phi_{ik} x_{ik} \right) \tag{11}$$

subject to

$$\sum_{i=1}^{n} x_{ik} \leq 1, \quad \forall k \in \{1,\dots,m\}$$

$$\sum_{k=1}^{m} x_{ik} \leq 1, \quad \forall i \in \{1,\dots,n\}$$

$$x_{ik} \in \{0,1\}, \quad \forall i,j.$$

Algorithm 1 Auctioneer robot i (b_i)

1: **for all** $k \in \{1,\dots,m\}$ **do**
2: $\quad \phi_{ik} = -V_i^k(b_i)$ {; Local bids}
3: $\quad$ Send ϕ_{ik} to neighbors.
4: **end for**
5: Receive bids from neighbors.
6: $\Phi = \{\phi_{jk}\}_{j,k}$ {; Create cost matrix}
7: $\{x_{jk}\}_{j,k} \leftarrow Hungarian(\Phi)$
8: **return** Policy selected for robot i (c_i).

In the execution phase, the best behavior for each robot is selected online with an auction algorithm (Spaan et al., 2010a) where the cost or bid of assigning a policy k to a robot i is $\phi_{ik} = -V_i^k(b_i)$. Thus, policies with a greater expected reward are more likely to be selected for each robot, which helps to maximize the global expected reward for the whole team. As explained in Section 4, the assignment of the behaviors can vary from one time step to the following in an uncorrelated manner. Note that these behaviors are not finite tasks that the robots must select and solve, but different policies to follow given the current belief. As the belief changes, the robots are allowed to switch their behaviors in order to pursue the optimal solution.

In the case $n > m$, the algorithm above will leave robots with no policy assigned. Therefore, the assignment problem is repeated with these free robots until they all get a policy. In this case, some policies would be assigned to more than one robot at the same time. Even though the existing behaviors are distributed equally among the teammates here (i.e. there are similar number of robots executing each behavior), the approach is more general. As it will be discussed in Section 5.3, alternative methods for role allocation could be used.

Algorithm 1 summarizes a decentralized auction in which the assignment problem is solved locally at each robot with the information available. Each robot i computes its own bids for the behaviors from its local belief b_i and communicates them to other neighboring robots. Then,

with the bids received from other robots, a local solution for the assignment problem (11) is obtained. This decentralized behavior assignment is solved at each decision step for the robots. The computation can be performed efficiently in polynomial time using the Hungarian algorithm (Burkard, 2002), a well-known algorithm for solving the assignment problem optimally. It computes a cost-minimizing assignment, operating on an $n \times m$ cost matrix, whose entries consists of ϕ_{ik} values. Hence, the allocation of behaviors to robots is optimal within each local auction.

The local cost matrices, and hence the local solutions for the behavior assignment, should be the same at each robot as long as all of the robots communicate with each other, and the communication is error-free. In that case, all of the local beliefs are equal and each robot creates a complete cost matrix with inputs from all of the robots in the team. However, if a robot only receives information from a subset of neighbors (due to communication failures or network topology), it can still solve the assignment with a cost matrix that only has inputs from those robots.

Owing to this partial information at each robot, or due to the fact that local beliefs for DDF systems are not synchronized all of the time, inconsistencies leading to suboptimal assignments may be obtained. In an inconsistent assignment, the local cost matrices and hence the local solutions differ from one robot to another. Therefore, a good synchronization of the local beliefs is desirable to avoid these suboptimal situations. In contrast, the robustness of the system is high, since information from all of the robots is not required to compute each local solution. In case some communication links failed, each robot would still get a suboptimal solution with the available information from their neighbors (robots from whom information was received). Moreover, the computation time of the Hungarian algorithm is relaxed, since each robot only runs it with a submatrix of the complete cost matrix.

In addition, there is another potential desynchronization when the robots make decisions at different moments and hence with different available information. Some previous works (Choi et al., 2009) propose consensus algorithms over this information in order to guarantee convergence for decentralized auctions even in case of time desynchronization. However, the decision-making performance is still degraded. Moreover, in the approach presented here robots are allowed to change their policies at every step. Therefore, the establishment of a previous consensus to converge to the same distributed solution is not worthwhile.

Furthermore, it is important to remark that, although independent POMDPs are solved for the robots, transition independence for the local states is not assumed. We are approximating a role-based MPOMDP and, hence, a joint belief is required for the team. This belief over the joint state and containing information from all of the robots, is provided by the DDF algorithm running during the execution phase. Nonetheless, to compute the bids for the auctions, the value functions of the different behaviors (defined over local states) have to be evaluated. The joint state variables can be marginalized out to keep only the local state variables and obtain b_i for each robot i.

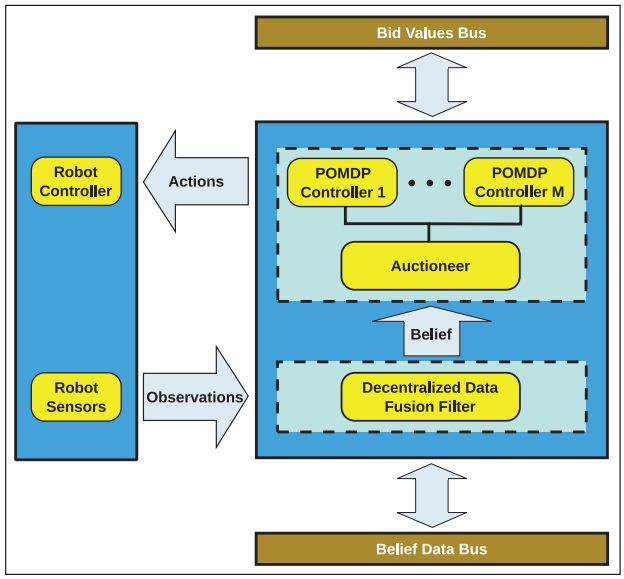

Fig. 5. Functional scheme for decision-making and belief update at each robot.

Finally, note that from the joint belief obtained through the DDF it is possible to predict locally the bids from other teammates, even if they do not communicate. Therefore, it may be possible to reason about other teammates' actions even in the case of no communication (in heterogeneous teams, robots could also require additional information about the capabilities of their teammates). Of course, the longer the communication breakdown lasts, the larger the difference with the assignments that would be obtained when additional information is received from other robots. In our current implementation, we just consider the robots within communication range to solve the behavior assignment, in order to have the most up-to-date information.

5.2. System overview

Figure 5 illustrates the system elements per robot. The whole process is separated into two different modules. Each robot can execute a certain number of behaviors modeled as single-robot POMDP controllers. A DDF module is in charge of computing the belief and feeding the Auctioneer module, which then chooses the adequate POMDP controller and the associated action. Even though most POMDP-based systems synchronize belief update and decision-making in the same loop, here the two processes are separated. In this way some constraints that limit the flexibility and robustness of the system are avoided. For instance, communication channels and transmission rates are totally independent for both modules, which is critical in decentralized systems under possible communication failures.

The approach is completely decentralized, since the belief estimation and the decision-making are carried out without the need for a central entity. On the one hand, the belief estimation is computed by a DDF algorithm that is distributed along the multiple robots. On the other hand, the POMDP controllers act also separately for each robot. Despite the fact that a multi-robot POMDP for the whole team is not solved (with its computational benefits), cooperative behavior still arises in two manners. First, the robots select their best role by explicitly reasoning about which behaviors their neighbors are performing via the auction mechanism. Second, the DDF also allows the robots to obtain information from further teammates. As explained in Section 2.3, the information travels throughout the network, and even robots that were not in direct communication range will recover a common belief eventually. Therefore, the joint beliefs contain implicit information from all of the teammates and help the robots to coordinate in an indirect manner.

5.3. Discussion

The decentralized auction proposed in this section provides a joint policy for the team $\pi' = \{\pi'_1, \ldots, \pi'_n\}$ that is suboptimal in the sense that does not optimize the original value function (10) for the role-based MPOMDP. Instead, the value that would be obtained after executing the computed policy π' is the following:

$$V'^{(t)}(b) = \sum_{s \in S} [R_1^{c_1}(s_1, a_1(\pi'_1)) + \cdots + R_n^{c_n}(s_n, a_n(\pi'_n))]b(s) + \gamma \sum_{z \in Z} p(z|a,b)\, V'^{(t-1)}(b_a^z) \quad (12)$$

$$with \quad \underset{\forall i \in \{1,\ldots,n\}}{c_i} = \arg\max_{c_i} \left[\sum_{j \in n_\alpha(i)} \sum_{k=1}^{m} V_j^k(b_j) \right]$$

where the optimizations for the behavior assignments are repeated at every decision step. Note that the role of each robot c_i is computed considering only its set of neighbors $n_\alpha(i)$ and their local beliefs. Analyzing (12), it can be seen more clearly how our method approximates the optimal policy for the role-based MPOMDP mainly in two manners:

1. The optimization process does not consider the joint reward term R^{joint} explicitly. Moreover, for assigning behaviors, the best long-term option is considered assuming that each robot will keep the same behavior during the whole time horizon. Thus, $\{c_i\}$ can be computed using the value functions from the individual behaviors.
   The fact that R^{joint} is not considered allows us to compute policies disjointly for single robots, which alleviates dramatically the complexity of the problem and makes it scalable with the number of robots. Moreover, to compensate for the lack of global optimality, $\{c_i\}$ are recomputed each time step during the execution of the policies, what allows the assignments to evolve suboptimally during the performance of the mission.
2. In the planning phase, the local policies are not solved considering joint beliefs (joint actions/observations), since they assume a single robot. However, joint information from all of the robots (joint belief) is accessible during the execution phase (DDF) to evaluate the local value functions V_i^k.
   In this paper, point-based algorithms are used to compute the policies. Owing to this, only a limited set of beliefs is taken into account to compute each value function. Thus, the set of belief points used when executing the policies is, in general, different from the set of belief points used to compute such policies, since joint beliefs are richer in terms of information (joint actions/observations). Nonetheless, the error that is made when a value function (approximated by any point-based method) is evaluated in a belief point that was not used during its computation, is bounded (Pineau et al., 2006).

Despite these approximations, our method can still get an exact solution for the role-based model in Section 4 for some particular sub-domains. In particular, if the following conditions are met: (i) individual policies for the roles are solved optimally; (ii) the joint belief computed by the DDF incorporates all of the robots (also the auctions) and is consistent; (iii) the planning horizon is 1; (iv) there is no joint term in the reward; and (v) different roles are assigned to different robots.

The first condition is standard for most POMDP works, since optimal policies can only be obtained in very restricted domains. The second condition is not a strong restriction either since when communication channels present negligible delays (when compared with the DDF rate), the DDF can achieve after some time such a consistent belief even when all robots are not in direct communication range (the belief could be propagated through the different subnetworks). Moreover, if all of the robots are not within communication range, at least the auctions would be optimal for each subteam. Assumptions (iii) and (iv) are the strongest. Regarding (iii), as explained above, we assume that robots do not switch roles when planning to compute the costs (which would just be true for a horizon equal to 1), but later, we compensate for this effect by recomputing the $\{c_i\}$ at each execution step (online re-planning). Even though assuming fixed roles makes planning easier, it could be the case that a robot needs to change its role in the middle of a mission (because conditions have varied) to achieve optimal performance. We account for that with this online re-planning.

The other strong assumption is to eliminate the joint reward. However, the cooperative behavior encoded in R^{joint} can be emulated by designing the roles properly for a given application. In the literature, there is some work about how to split the joint problem into individual sub-problems or

roles. In particular, Matignon et al. (2012) design individual rewards towards a global objective trying to minimize the interactions among the robots. Moreover, Sleight and Durfee (2012) and the references therein contribute with theoretical discussion on how to design individual models for the teammates so that, put together, lead to joint cooperation.

Finally, condition (v) is not so restrictive. In this paper, we focus on a strategy in which the existing behaviors are assigned equally among the different robots in the team (Hungarian algorithm in (11)). This applies to many applications in which performing all of the available tasks in a homogeneous fashion is beneficial. However, our approach is more general and alternative methods to solve the role assignment could be possible. For instance, some robots could get no role assigned depending on the circumstances (they would stay in an idle status). Also, more complex constraints could be added to the problem, such as robots that can only play a subset of the roles (i.e. heterogeneous capabilities for the robots), or subsets of roles that must be assigned jointly (e.g. a couple of robots carrying something together).

Unfortunately, for the general case, obtaining meaningful bounds on the loss in value that our approximations incur is hard. In fact, by removing the synchronized belief assumption present in MPOMDP models, optimal policies can be computed using a role-based variation of the Dec-POMDP model. One such variation, the role-based multiagent team decision problem, has been shown to be NEXP-complete (Nair et al., 2003). Furthermore, computing an ϵ-approximate joint policy for a general Dec-POMDP is NEXP-complete as well (Rabinovich et al., 2003).

Regarding complexities, our approach scales far better with the number of robots than the MPOMDP. In the worst case, the complexity of a single step of value iteration (3) for the MPOMDP is $O(|S|^2|A||\Gamma_{t-1}|^{|Z|})$ (Pineau et al., 2006), where $|S|$, $|A|$ and $|Z|$ grow exponentially with the number of robots.[2] In the worst case, our approach solves $m \times n$ local POMDPs (one per robot and behavior), each of them with a maximum complexity of $O(|S_i|^2|A_i||\Gamma'_{t-1}|^{|Z_i|})$ for a single value iteration. Apart from computing the policies, we also solve the auction at each time step during the execution phase, which entails a complexity $O(\max(n, m)^3)$ in the worst case (Burkard, 2002).

In our approach, even though the total computation time increases with n and m, it does so polynomially instead of exponentially. Moreover, these values are usually bounded for real applications. On the one hand, the number of available behaviors is limited for most role-based real applications, and it depends on the problem itself. Also, in case of necessity the number of behaviors m could be traded off according to computational constraints. On the other hand, since the auctions are solved locally at each robot using only its available information, in practice, the robots only work with a subset of neighbors $n_\alpha \leq n$, that are those within communication range. Although it is not the focus of this paper, in the literature there exist other works that propose more efficient solutions for the assignment problem in case of large teams (Liu and Shell, 2011).

6. Case study 1: environmental monitoring with UAVs

In order to illustrate the flexibility and scalability of the proposed approach, two different applications with multiple robots are presented: environmental monitoring with UAVs and cooperative tracking. In this section the models and results of the first case are presented.

6.1. Scenario definition

In this problem there is a team of n UAVs whose mission is to fly over a certain terrain in order to monitor a potential contamination that may appear. It is assumed that this contamination can only appear and propagate through a network of water flows on the terrain. Therefore, instead of surveying the whole scenario, a set of key points within that network can be extracted to evaluate the level of contamination. These points are inter-connected through water flows and the contamination can propagate among them.

The scenario is discretized into a cell grid and it is assumed that the overall contamination can be controlled reliably by surveying a set of m critical cells. The objective of the team of UAVs is to decide how to visit the critical points optimally in order to reduce the joint uncertainty on the contamination level. Figure 6 depicts this scenario adapted to the National Park of Doñana, a remarkable marshland located in the south of Spain. Owing to the huge number of species that it hosts, contamination or any other natural disaster are real threats that need to be controlled.

The problem can be modeled using POMDPs. All of the details are given in the Appendix. In particular, each UAV is equipped with a camera sensor pointing downwards that provides a (noisy) binary observation about the contamination level of the cell in which it is located: *yes* or *no*. We assume that the UAVs are helicopters (such as this in Figure 1), so at each time step, they can *stay* (hovering) in the same cell or moving to a neighboring cell: *north*, *west*, *east* or *south*. Noisy transition functions are considered for these movement actions. In addition, when a UAV is on top of a critical point, instead of moving, it can select two additional actions (*classCont* and *classNotCont*) to classify that area as contaminated or non-contaminated, respectively.

There is a factored state with a set of variables describing the contamination level of each critical point, which can be: *none*, *low* or *high*. A graph describing the interconnections among the critical points (due to water flows) is also known. Thus, the evolution of the contamination is modeled so that it can start at certain points of the graph (entry points), and these effects can be propagated to the other inter-connected points downstream. In addition, there

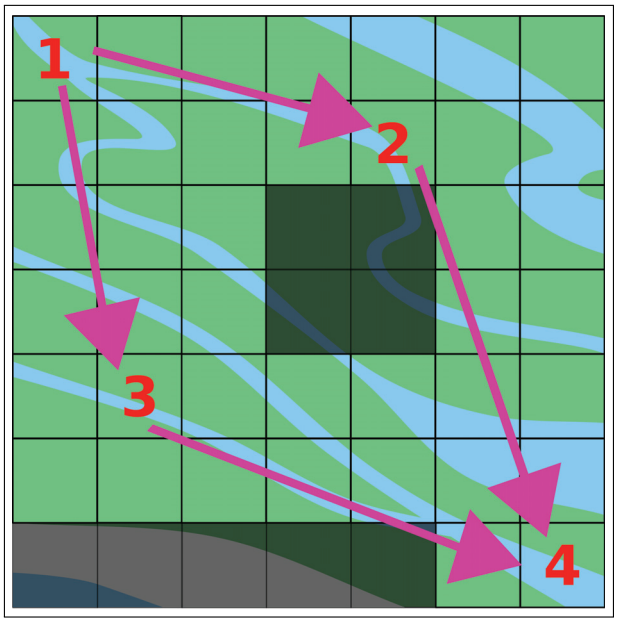

Fig. 6. Piece of marshland in the National Park of Doñana. The shadowed cells in the grid represent non-flying zones: a beach; and the core of the park. Each critical cell is marked with a number, and the propagation graph with arrows.

is another binary state factor for each critical point to specify whether it was already classified or not. Thus, if a UAV that is on top of a critical cell takes one of the two classification actions, the corresponding variable for that critical point is set to 1. Otherwise, if there are not classification actions but the critical point was previously classified, it can switch back to 0 (not-classified) with a probability p_{des} at each time step. This is to allow the critical areas to be declassified again after some time. The local state for each UAV consists of all of the mentioned factors and an additional one indicating its position on the grid.

In this application, the main objective is to reduce the uncertainty over the contamination level. This is done by monitoring the critical points and classifying them when their uncertainty is low enough. According to the classic POMDP formulation that is considered in this paper, information gain cannot be rewarded explicitly, since the rewards are state-based (rewards depending on entropies would be belief-based). However, when having better information improves the task performance (e.g. less uncertainty on an event detection reduces the risk of a wrong detection), the POMDP policy will try to select these information-gaining actions. Therefore, the idea here is to maintain state-based rewards (and, hence, a classic POMDP framework with linear value functions), but adding classification actions in order to reward UAVs for reaching a certain level of uncertainty regarding contamination.

According to this idea, positive rewards are obtained when right classification actions are taken, whereas negative rewards are obtained when the classifications are wrong. The resulting policy will lead to beliefs with low uncertainty on the contamination state, for which the UAVs are more likely to make right classifications.

The approach proposed in this paper can be used to solve this problem. A single-robot behavior is considered for each possible point to monitor (m behaviors). Thus, the reward function for each behavior k (R_i^k) rewards UAV i only if it classifies the critical area k.

6.2. Results

This case study has been analyzed in a simulated environment. In the following subsections, the setup for the different simulations and their corresponding results are detailed.

6.2.1. Simulation setup Experiments on environmental monitoring were performed on a simulated scenario (MATLAB) of the National Park of Doñana described above (see Figure 6). In order to survey the Park with a team of UAVs, it was divided into the 7×7 grid shown in Figure 6, where the dark shaded cells represent non-flying zones that the UAVs cannot access for security reasons. Since the contamination is assumed to propagate through the water flows, the four key areas shown in Figure 6 are used for surveillance. The inter-connection graph considered to model the propagation is also depicted in the figure. Moreover, it is assumed that contamination could only start at Area 1 (entry point).

Each cell in the grid is a square of 1×1 km and can be surveyed by a UAV whenever it is flying on it. For these experiments, the probability to declassify areas is set to $p_{des} = 0.04$. Moreover, due to the spatial resolution of the grid, the dynamics of the UAVs and the contamination model are slow. Therefore, a time step of 10 minutes is assumed in the simulations.

As mentioned in Section 5.2, the communication channels for the DDFs and the Auctioneers are independent. In these experiments, the UAVs communicate their DDF information in a tree-like topology, whereas the bids for the auctions are sent to all others in the neighborhood (the required bandwidth for the latter is not significant).

There are 4 different behaviors, one for each critical area. A single-robot policy was computed for each of them with a Java version of Symbolic Perseus.[3] The solver ran 10 minutes for each policy in a computer with an Intel Core processor (4 cores @2.67GHz) and 8 GB RAM.

The complexity of the models considered is summarized in Table 1. It is important to highlight that, in order to alleviate the complexity of the belief space, mixed observability Markov decision processes (MOMDPs) (Ong et al., 2009) were considered to find the policies for all of the experiments in this paper. The robots' positions were assumed to be observable within the POMDP, which is reasonable if the sensors for self-positioning are accurate enough for the given grid resolution.

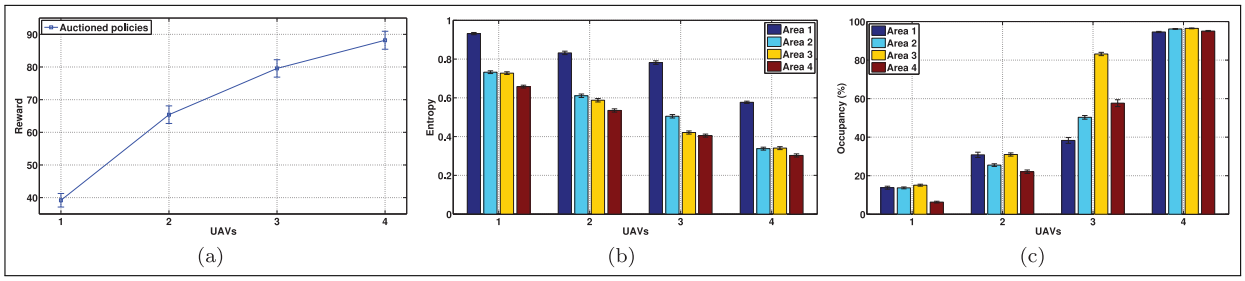

Fig. 7. Simulations of an environmental monitoring application with four critical areas. The average results ($\pm 3\sigma$) for auctioned policies are shown varying the number of UAVs involved. (a) Discounted rewards. (b) Entropies of the beliefs on the contamination levels. (b) Percentages of occupancy for each critical area.

Table 1. Complexity of the POMDP models used in this paper. Number of states (the observable states are underlined), actions and observations are computed for the general multi-robot case and for a single-robot case.

	$\|\mathbf{S}\|$	$\|\mathbf{A}\|$	$\|\mathbf{Z}\|$
Monitoring (n robots, 4 areas)	$\underline{40^n} \times 81 \times 16$	7^n	2^n
Monitoring (1 robot, 4 areas)	$\underline{40} \times 1{,}296$	7	2
Tracking (n robots)	$\underline{82^n \times 4^n} \times 82$	4^n	2^n
Tracking (1 robot)	$\underline{328} \times 82$	4	2

6.2.2. Simulations We tested our approach with auctioned policies for teams with 1, 2, 3 and 4 UAVs, each of them running an estimation filter implementing the DDF scheme in Section 2.3, and an auctioneer controller that executed the algorithm in Section 5.1. For each team, 1,000 simulations of 100 steps were performed with the UAVs starting at random positions. Moreover, all of the simulations started without contamination, but there was a probability of 0.1 that contaminated water appeared at the entry point (Area 1) at any moment.

The average discounted rewards and belief entropies for all the experiments are shown in Figure 7a and (b), respectively. It can be seen how the addition of more UAVs improves the performance, increasing the reward of the team and decreasing the entropy of the belief on the contamination levels for each area ($\sum_{\forall level} -p_{level} \log(p_{level})$). Note that the entropy of Area 1 is always higher, since there is the uncertainty of new contamination appearing. In Figure 7c, the percentage of time that each area is visited by any UAV is also shown. The more UAVs there are, the better they can cooperate to cover all of the areas.

Since only single-robot policies are computed in our approach, the complexities of the models do not increase with the number of UAVs (see Table 1), which makes the solution scalable. Nonetheless, in this scenario, experiments with more than four UAVs are not presented because they do not improve the performance significantly (four UAVs can already cover all of the critical areas). Moreover, it may look like the propagation model does not scale with the number of areas. However, if the interconnection graph for the areas is sparse enough, we can still keep bounded the number of parents for each area, and just increase moderately the complexity of the factored model.

We also tested our approach against a joint policy for a multi-robot POMDP. The multi-robot POMDP is far from scalable (see Table 1), so we were only able to solve it for a simple case with 2 UAVs and 3 areas (Areas 1, 2 and 3).[4] Actually, any variation of this small scenario considering more UAVs or areas, caused the same computer mentioned above to run out of memory.

We used Symbolic Perseus[5] again to compute a single-robot policy for each behavior (5 minutes each) and a joint policy for the 2-UAV MPOMDP (14 hours). Then, we ran 1,000 simulations of 100 steps (with random starting positions and no initial contamination) for our approach, and the same with the joint policy. The average values for the belief entropies and the percentage of occupancy (times visited) of each area are shown in Figure 8. Despite the huge difference in computational time for both approaches, the results are still remarkably similar. Of course, the joint policy should be better for more complex examples, but its computation becomes intractable. Furthermore, the average discounted rewards were 53.4970 ± 0.91 for our approach, and 41.0252 ± 1.08 for the joint policy. The reward function used to compute the joint policy is used to evaluate both approaches. In this reward function, apart from the local rewards for each UAV, there is also a joint reward term, which does not allow two UAVs to get rewards simultaneously for classifying the same area (see the Appendix for details). In this case, our approach outperforms the joint policy, which is possible given that all of the policies are approximate.

7. Case study 2: multi-robot cooperative tracking

This section describes the case study about cooperative tracking. The models for this application are explained and the corresponding experiments detailed.

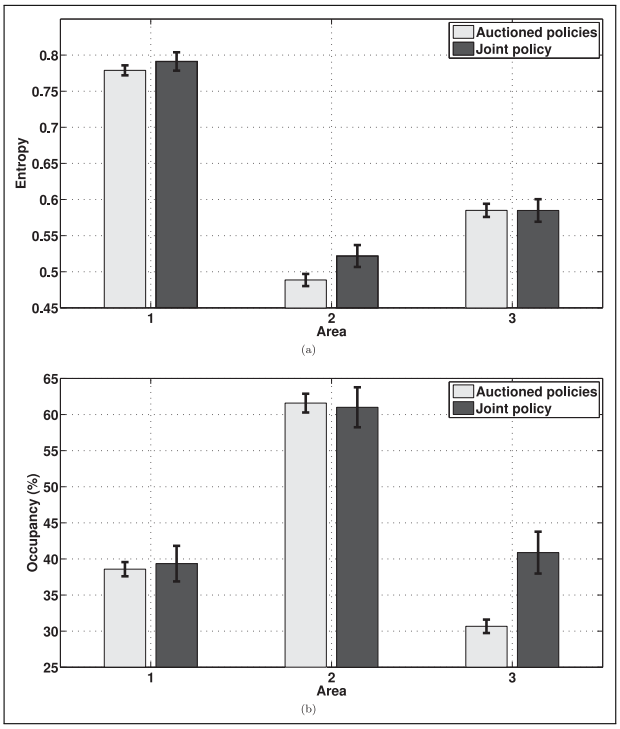

Fig. 8. Average results ($\pm 3\sigma$) for simulations on environmental monitoring with two UAVs and three critical areas. Auctioned polices are compared with a joint policy. (a) Entropies of the beliefs on the contamination levels. (b) Percentages of occupancy for each critical area.

7.1. Scenario definition

In this case, the objective is that a group of n robots track a moving target estimating its position with their sensors. The idea is to obtain an estimation as accurate as possible. Target tracking problems benefit from reasoning about future steps (He et al., 2010). Besides, cooperative behaviors are particularly helpful when there are multiple robots involved in the tracking.

To model this scenario as a POMDP, the local state for each robot is composed of the position of the target and its own position and heading. The state space is discretized into a cell grid, and a map of the scenario is assumed to be known. All of the details about the models are shown in the Appendix. In particular, there are four possible headings for each robot: *north*, *west*, *south* or *east*.

At each time step, the robots can choose between four possible actions: *stay*, *turn right*, *turn left* or *go forward*. *stay* means doing nothing; when *turning*, the robot changes its heading 90°; and when *going forward*, it moves to the cell ahead. Nonetheless, noisy transition functions for the states of the robots are considered. In addition, the target is assumed to move randomly. Therefore, the transition function for its position indicates that from one time step to the next, the target can move to any of its eight-connected cells with equal probability (only non-obstacle cells are considered in order to calculate that probability).

In addition, the robots carry a sensor that provides a boolean measurement: *detected* or *non-detected*. These sensors proceed as follows, if the target is out of its field of view (FOV), the sensor produces a *non-detected* measurement. However, when the target is within its FOV, it can be *detected* with a probability p_D.

The robots aim to reduce the uncertainty of the target estimate. The FOV of their sensors (a 3×4 rectangle in front of the robot), which is shown in Figure 9b, entails mainly uncertainty in depth, since the heading where the target is can be determined precisely. Pointing at it from different angles definitely helps to reduce the uncertainty of its estimate by achieving a partial overlapping in the FOVs. Therefore, cross configurations should be fostered by giving a high reward to each robot that is keeping the target within its FOV, and even higher if the robot's orientation differs from the others'.

The method presented in this paper to combine individual behaviors is used. Since robots should cooperate to track the target from different directions, each behavior consists of following the target from a specific direction. Here, a single-robot behavior for each possible orientation is considered: $\{north, west, south, east\}$. Therefore, the reward function for the behavior k (R_i^k) gives a high reward to robot i only if the target is within its FOV and the robot's heading corresponds to the orientation of behavior k. Also, since the objective is tracking the target, the robots should position their sensors in the best way not to lose it. For the sensors proposed, the high reward is only obtained when the target is in one of the closest cells. The cells with high rewards are illustrated in Figure 9b.

7.2. Results

Real and simulated experiments on cooperative tracking were conducted with a real testbed[6] that allows the user to combine simulated and real robots (Pioneer-3AT), an illustration of which is shown in Figure 9a. The simulated version ran with Player/Stage.

7.2.1. Experimental setup The map of this testbed was discretized into 2 m $\times$ 2 m cells and resulted in the occupancy grid of 12×10 dimensions shown in Figure 9b, where cells representing obstacles are in black. A team of robots is considered in order to follow a target, represented by another robot. All of the pursuers present similar perception capabilities by means of a sensor with $p_D = 0.9$ and the FOV shown in Figure 9b. Note that the pursuer observations were obtained by simulating sensors with the mentioned capabilities on board the robots, since the development of real detectors is out of the scope of this paper.

During all of the experiments the target followed a path unknown for the pursuers and with a random component. A path planning algorithm was used to obtain the path to the high-level goals provided by the POMDP controllers (next cell to move and robot heading), whereas a local navigation

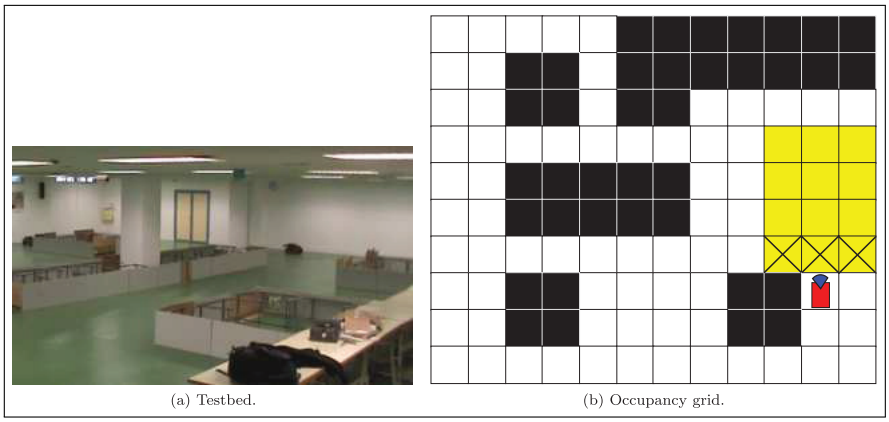

Fig. 9. (a) Picture of the multi-robot testbed. (b) Testbed occupancy grid (black cells are obstacles) and an example of the FOV for a robot (yellow cells, 3 × 4 rectangle in front of the robot). All of the robots have the same FOV. In addition, if the target is in one of the cells with crosses (closest part of the FOV) and the heading is adequate, a high reward is obtained.

algorithm was used to safely navigate the given path. Each robot had running onboard an implementation of the DDF (Section 2.3) and an auctioneer controller (Section 5.1). Again, as in the previous case study, the DDFs communicate in a tree-like topology, whereas the bids for the auctions are sent to all others in the neighborhood.

Three different approaches were tested: (i) auctioned POMDPs with DDF; (ii) auctioned POMDPs without DDF; (iii) independent POMDPs with DDF. The two first approaches are based on the auction method proposed in this paper, but in the second one, neither communication nor fusion is considered for the DDF modules. In the third approach, a single and independent POMDP is used for each robot and communication between the DDF modules is allowed. Moreover, all of the policies were obtained by solving the corresponding MOMDPs with a C++ implementation of the SARSOP algorithm (POMDP complexities are shown in Table 1). The solver ran 1,700 seconds for each policy in a computer with an Intel Core 2 Duo processor @2.47GHz and 2.9 GB RAM. For the approaches (i) and (ii), a different MOMDP is solved for each heading, whereas for approach (iii), there is a single MOMDP independent of the heading (no roles).

7.2.2. Real robots First, some experiments[7] were carried out in order to compare our approach with the others mentioned above. Three of the real Pioneer-3AT were used to track the remaining one, that played the target's role. In order to have similar conditions for each run, the three robots always started at the same fixed points and the sample times were the same, 10 seconds for the decision-making and 3 seconds for the DDF. An experiment of 15 minutes was performed for each of the three approaches.

Some average results with their standard deviations are presented in Table 2. At each time step, the target location is estimated by searching the cell of the belief with a highest likelihood. This value is compared with the actual target position. The entropies of the belief at each step

Table 2. Average results of the experiments with a three-robot team for three different approaches. The error of the estimated position of the target with respect to its actual position and the entropy of the estimated beliefs are shown.

	Error(m)	Entropy
	Auction+DDF	
Robot 0	4.07 ± 0.16	2.61 ± 0.05
Robot 1	3.95 ± 0.15	2.55 ± 0.05
Robot 2	4.18 ± 0.16	2.66 ± 0.05
	Independent+DDF	
Robot 0	6.86 ± 0.32	2.80 ± 0.05
Robot 1	6.70 ± 0.32	2.70 ± 0.05
Robot 2	6.75 ± 0.32	2.68 ± 0.05
	Auction	
Robot 0	9.74 ± 0.29	3.85 ± 0.03
Robot 1	9.41 ± 0.34	3.28 ± 0.06
Robot 2	10.46 ± 0.40	3.62 ± 0.04

($\sum_{\forall cell} -p_{cell} \log(p_{cell})$) are also averaged and shown. It can be seen that the approach proposed in this paper (Auction+DDF) reduces the entropy and the target localization error with respect to the independent POMDPs approach, since the cooperation between the robots allows them to track the target from different points of view. It can also be noticed that the estimation of the target position is worse for the auctioned approach in which no DDF is included. Note that in this case, the mean errors are bounded by the resolution of the cells (2 m). The option of independent POMDPs without DDF resulted in very poor performance (and, hence, is not included), as the robots do not share any information nor coordinate their behaviors.

Owing to the bearing information encoded in the sensor observations, a cross-configuration among the pursuers allows them to point at the target from different points of view and reduce the uncertainty of its estimation. This cross-configuration is fostered by our auctioned approach,

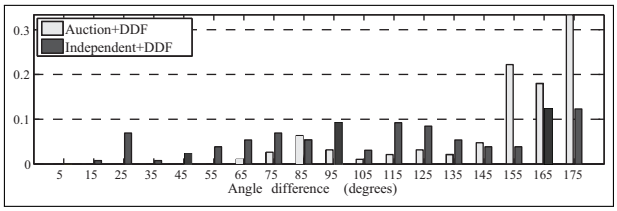

Fig. 10. Normalized histograms of the maximum angle differences between the robots when the target is within the FOV of any of them.

as it can be seen in Figure 10. This figure compares our approach to the independent POMDPs with DDF in terms of angle configuration between the pursuers. Normalized histograms of the maximum angle difference between any of the pursuers every time the target is within FOV are shown. The Auction+DDF histogram presents a high peak close to 180° and a small mode in 90° (cross configurations), whereas the histogram is quite flat for the independent POMDPs. This shows that the proposed approach is more effective in reaching cross-configurations.

Second, to show the scalability and robustness of the system, a tracking experiment with a four-robot team was performed. In this case, the target was represented by a simulated robot (we only had four robots available). We were able to run this experiment for more than 30 minutes with the algorithms working on board the robots in a distributed way and using WiFi communications, showing the robustness of the system. An extract of the trajectories followed by the pursuers and the target can be seen in Figure 11a. The orientation of the pursuers at the end of the experiment has also been plotted to show how they surround the target to reduce the estimation uncertainty. Note that, when the target turns right, since they know the map, the pursuers opt for going directly to the other exit of the aisle so they can find it there.

The cooperation between the members of the team is depicted in Figure 11b, which shows the policies allocated to each robot during the same time frame (each iteration takes place every 10 seconds). Owing to differences in the local beliefs and different decision times for the robots, inconsistent solutions (robots with the same policy) are obtained in some occasions. However, as time passes the robots have built up a better belief regarding the target's position, and the assignment stabilizes (after iteration 22 in Figure 11b). Information regarding the time spent solving the auctions will be provided in the next section.

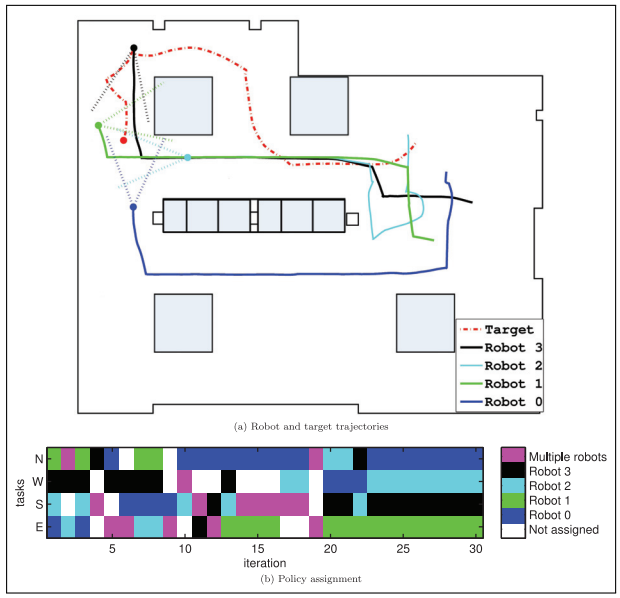

Fig. 11. Experiment with a four-robot team. (a) Trajectories followed by the robots and the target during the experiment. Orientations at the last time step are also shown. (b) Policies allocated to each robot during the same time frame.

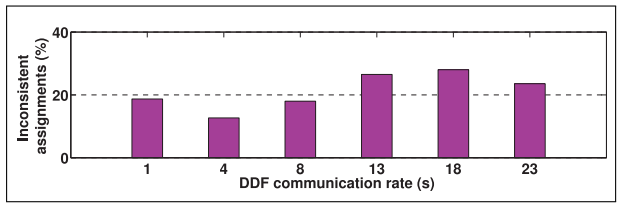

Fig. 12. Evolution of the performance of the auction under variable transmission rate for the DDFs.

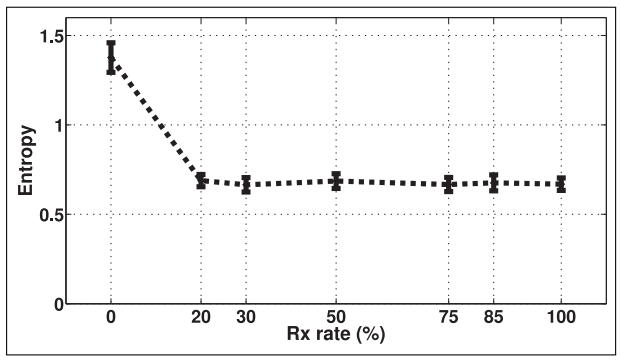

Fig. 13. Average entropy ($\pm 3\sigma$) of the target estimate depending on the rate of received bids. Some bids are not received to simulate noisy communication channels.

7.2.3. Simulations Some simulations were run in the simulated version of the testbed. First, simulations to check how our auction algorithm performs when robots do not access the same information. The experiments consisted of three simulated robots, two pursuers and a target, starting at the same positions in each experiment and with sample times as before. Communication latency for the DDF modules was varied throughout the simulations (two simulations of 20 minutes for each latency value). Communication rates for the bid values were maintained, as the data volume is not significant for the bandwidth compared with belief information. The performance of the distributed auction algorithm is shown in Figure 12 by representing the

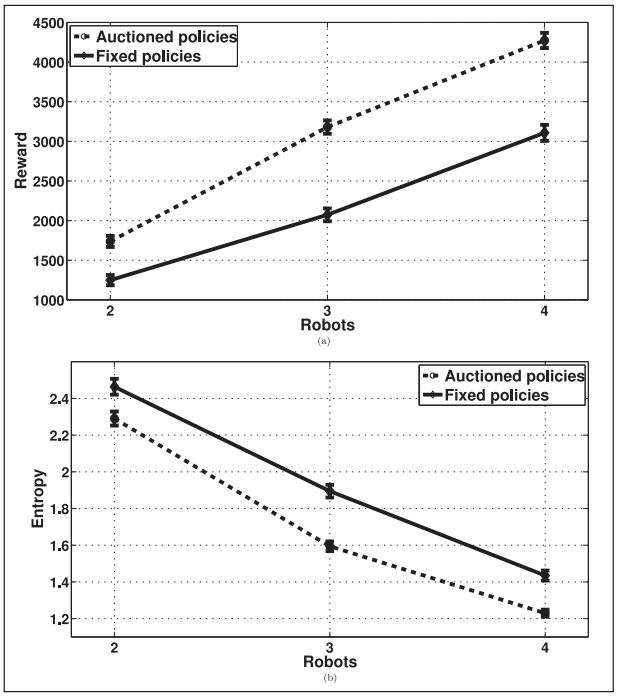

Fig. 14. Average results ($\pm 3\sigma$) for simulations on cooperative tracking with different numbers of robots involved. Auctioned policies are compared with fixed policies assigned to the robots. (a) Discounted rewards. (b) Entropies of the belief on the target position.

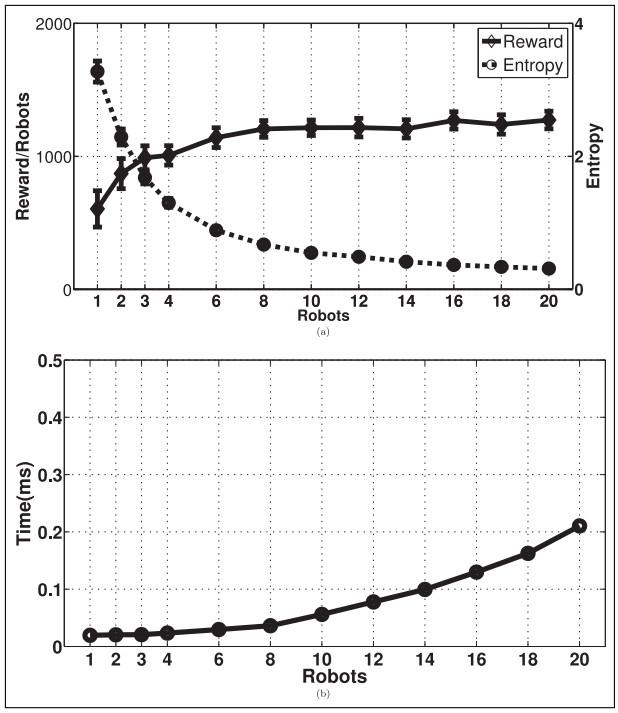

Fig. 15. Average results ($\pm 3\sigma$) on cooperative tracking to show scalability with multiple robots. (a) Entropy of the target estimation and discounted reward per robot. (b) Average time spent to solve a role assignment at each robot.

percentage of inconsistent assignments. As the communication rate for the DDFs is increased, the difference between the beliefs to which the robots have access grows too. Thus, the consistency of the assignments becomes more difficult under worse communications. However, Figure 12 shows graceful degradation with respect to the communication rate.

Second, an experiment with eight robots simulating noisy communication channels was performed. In this experiment the robots were accessing the joint belief, but a percentage of the packages with bid values was lost during communication. Again, a graceful degradation in the entropy of the target estimate is shown in Figure 13 as we reduce the percentage of bids received successfully (100 runs of 100 steps were carried out for each rate value). For comparison, we added the worst possible case, in which all of the bids were lost and each robot selected its role without any reasoning about the others.

Third, we ran simulations in order to prove the better performance of our auctioned algorithm against a naive approach in which the same fixed behavior was assigned to each robot during the whole experiment. With the robots starting always at the same positions, located at the four cardinal points of the scenario, 1,000 runs of 100 steps were carried out considering 2, 3 and 4 robots. In the case of fixed policies, each robot was always assigned the one that fitted better according to its starting position (e.g. the robot starting at the north was commanded to approach the target from the north). In Figure 14, the average discounted rewards and entropies on the target estimation are shown together with their standard deviations. It can be seen that for all of the cases the auction approach performs better, since the robots are able to adapt to different situations throughout the experiment.

Finally, some simulations were run in order to provide empirical results about scalability. We ran our approach in the same scenario but increasing the number of robots in the team. For each experiment, 100 runs of 100 steps were carried out with the robots and the target starting at random positions. The entropy of the target estimation and the total average reward (discounted) divided by the number of robots are shown in Figure 15a. It can be seen how the performance does not increase significantly after six out of eight robots. This is due to the size of the scenario, which can be handled with a few robots. In Figure 15b, the average time spent at each robot (per iteration) to solve the role assignment is plotted. In this case, all of the robots were considered within communication range in order to evaluate the worst possible case (auctions needed to be solved for the complete team). It can be seen that the execution time scales polynomially with the number of robots, but in general, our approach will only depend on the number of neighbors, which should be lower than the team size for larger scenarios. Nonetheless, even in this worst case, execution times stay at quite reasonable levels for real-time applications.

8. Conclusions

Planning-under-uncertainty techniques, such as POMDPs, face a scalability problem when considering teams of robots, becoming intractable quite easily. Popular frameworks such as Dec-POMDPs scale poorly to many robots, unless very severe independence assumptions are applied (Nair et al., 2004). Furthermore, many of these models either do not allow robots to exploit inter-robot communication, or implicitly assume instantaneous cost-less communication (MPOMDP). We focus on scalable techniques that do not require such strict communication guarantees, which are hard to meet in multi-robot domains with unreliable wireless channels.

For certain role-based applications, we approximate the MPOMDP solution by auctioning independent POMDP-based controllers in a cooperative fashion. We also relax the communication guarantees by introducing a DDF approach for belief propagation, which allows for imperfect communication channels and makes the system more reliable. Even though the solution of our approach is suboptimal, the results obtained in terms of cooperative behavior are still good. Moreover, since the computational complexity is reduced dramatically, it is much more scalable than other multi-robot POMDP approaches, offering a trade-off between optimality and applicability.

We present results on environmental monitoring and cooperative tracking applications that cannot be solved with the current state of the art in multi-robot POMDP solvers. In addition, there are many other multi-robot applications that can be modeled with cooperative roles and solved with our framework.

In the future, more research is still necessary to evaluate the exact degradation that we suffer against optimal solutions. Also, some methods to analyze the initial problem and identify potential sets of roles would be of interest. So far, those roles are set in an *ad hoc* fashion.

Funding

This work was partially funded by Fundação para a Ciência e a Tecnologia (ISR-IST) (project number CMU-PT/SIA/0023/2009), the FP7 Marie Curie Actions Individual Fellowship (number 275217 (FP7-PEOPLE-2010-IEF)), the FP7 EC-SAFEMOBIL Project (grant number 288082) and the PAIS-MultiRobot (TIC-7390) Regional Project.

Notes

1. For the sake of clarity, in the rest of the paper s, a and z refer to joint variables whereas local variables are indicated with sub-indices.
2. In general, $|S| = \prod_{i=1}^{n} |S_i|$, and the same applies to $|A|$ and $|Z|$.
3. The parameters for Symbolic Perseus were 5,000 belief points, 1,500 α-vectors maximum, 30 iterations per round and 5 rounds.
4. The MPOMDP was designed to reward only one UAV at a time in case of several classifying the same area. This fostered the distribution along the different critical points.
5. The parameters for Symbolic Perseus were 5,000 belief points, 700 α-vectors maximum, 10 iterations per round and 5 rounds.
6. See http://www.cooperating-objects.eu.
7. See video in Extension 1.

References

Bernstein DS, Givan R, Immerman N and Zilberstein S (2002) The complexity of decentralized control of Markov decision processes. (d) *Mathematics of Operations Research* 27: 819–840.

Bourgault F and Durrant-Whyte H (2004) Communication in general decentralized filters and the coordinated search strategy. (d) In: *Proceedings of the 7th International Conference on Information Fusion*.

Boutilier C (1996) Planning, learning and coordination in multiagent decision processes. (d) In: *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge*, pp. 195–210.

Burdakov O, Doherty P, Holmberg K, Kvarnstrom J and Olson PM (2010) Relay positioning for unmanned aerial vehicle surveillance. (d) *The International Journal of Robotics Research* 29(8): 1069–1087.

Burkard RE (2002) Selected topics on assignment problems. (d) *Discrete Applied Mathematics* 123(1–3): 257–302.

Capitan J, Merino L, Caballero F and Ollero A (2011) Decentralized delayed-state information filter (DDSIF): A new approach for cooperative decentralized tracking. (d) *Robotics and Autonomous Systems* 59: 376–388.

Choi HL, Brunet L and How J (2009) Consensus-based decentralised auctions for robust task allocation. (d) *IEEE Transactions on Robotics* 25: 912–926.

Cole D, Sukkarieh S and Goktogan A (2006) System development and demonstration of a UAV control architecture for information gathering missions. (d) *Journal of Field Robotics* 23(6–7): 417–440.

de Hoog J, Cameron S and Visser A (2009) Role-based autonomous multi-robot exploration. (d) In: *Proceedings of COMPUTATIONWORLD '09*, pp. 482–487.

Doshi F and Roy N (2008) The permutable POMDP: fast solutions to POMDPs for preference elicitation. (d) In: *Proceedings of AAMAS*, volume 1, pp. 493–500.

Foka A and Trahanias P (2007) Real-time hierarchical POMDPs for autonomous robot navigation. (d) *Robotics and Autonomous Systems* 55(7): 561–571.

Gerkey BP and Matarić MJ (2004) A formal analysis and taxonomy of task allocation in multi-robot systems. (d) *The International Journal of Robotics Research* 23(9): 939–954.

Gmytrasiewicz PJ and Doshi P (2005) A framework for sequential planning in multi-agent settings. (d) *Journal of Artificial Intelligence Research* 24: 49–79.

He R, Bachrach A and Roy N (2010) Efficient planning under uncertainty for a target-tracking micro-aerial vehicle. (d) In: *Proceedings of ICRA*, pp. 1–8.

He R, Brunskill E and Roy N (2011) Efficient planning under uncertainty with macro-actions. (d) *Journal of Artificial Intelligence Research* 40: 523–570.

Hoey J and Little JJ (2007) Value-directed human behavior analysis from video using partially observable Markov decision processes. (d) *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29(7): 1–15.

Hsiao K, Kaelbling L and Lozano-Perez T (2007) Grasping POMDPs. (d) *Proceedings of ICRA*, pp. 4685–4692.

Hsieh MA, Cowley A, Keller JF, et al. (2007) Adaptive teams of autonomous aerial and ground robots for situational awareness. (d) *Journal of Field Robotics* 24: 991–1014.

Kaelbling LP, Littman ML and Cassandra AR (1998) Planning and acting in partially observable stochastic domains. (d) *Artificial Intelligence* 101: 99–134.

Kok JR, Spaan MTJ and Vlassis N (2005) Non-communicative multi-robot coordination in dynamic environments. (d) *Robotics and Autonomous Systems* 50(2–3): 99–114.

Kurniawati H, Du Y, Hsu D and Lee WS (2011) Motion planning under uncertainty for robotic tasks with long time horizons. (d) *The International Journal of Robotics Research* 30(3): 308–323.

Kurniawati H, Hsu D and Lee W (2008) SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. (d) In: *Proceedings of the Robotics: Science and Systems Conference*. Zurich, Switzerland.

Leonard NE, Paley DA, Davis RE, Fratantoni DM, Lekien F and Zhang F (2010) Coordinated control of an underwater glider fleet in an adaptive ocean sampling field experiment in Monterey Bay. (d) *Journal of Field Robotics* 27: 718–740.

Liu L and Shell DA (2011) Multi-level partitioning and distribution of the assignment problem for large-scale multi-robot task allocation. (d) In: *Proceedings of RSS*.

Matignon L, Jeanpierre L and Mouaddib A (2012) Coordinated multi-robot exploration under communication constraints using decentralized Markov decision processes. (d) In: *Proceedings of AAAI*, pp. 2017–2023.

Maza I, Caballero F, Capitan J, de Dios JM and Ollero A (2011) A distributed architecture for a robotic platform with aerial sensor transportation and self-deployment capabilities. (d) *Journal of Field Robotics* 28(3): 303–328.

Merino L, Caballero F, de Dios JM, Ferruz J and Ollero A (2006) A cooperative perception system for multiple UAVs: application to automatic detection of forest fires. (d) *Journal of Field Robotics* 23: 165–184.

Merino L, Gilbert A, Capitan J, Bowden R, Illingworth J and Ollero A (2012) Data fusion in ubiquitous networked robot systems for urban services. (d) *Annals of Telecommunications* 67(7–8): 355–375.

Mosteo A and Montano L (2007) Comparative experiments on optimization criteria and algorithms for auction based multi-robot task allocation. (d) In: *Proceedings ICRA*, pp. 3345–3350.

Nair R, Tambe M and Marsella S (2003) Role allocation and reallocation in multiagent teams: towards a practical analysis. (d) In: *Proceedings AAMAS*, pp. 552–559.

Nair R, Tambe M, Roth M and Yokoo M (2004) Communication for improving policy computation in distributed POMDPs. (d) In: *Proceedings AAMAS*, volume 3, pp. 1098–1105.

Nair R, Varakantham P, Tambe M and Yokoo M (2005) Networked distributed POMDPs: a synthesis of distributed constraint optimization and POMDPs. (d) In: *Proceedings AAAI*, pp. 133–139.

Nettleton E, Durrant-Whyte H and Sukkarieh S (2003) A robust architecture for decentralised data fusion. (d) In: *Proceedings of ICRA*.

Oliehoek FA and Spaan MTJ (2012) Tree-based pruning for multiagent POMDPs with delayed communication. (d) In: *Proceedings AAAI*, pp. 1415–1421.

Ong S, Png SW, Hsu D and Lee WS (2009) POMDPs for robotic tasks with mixed observability. (d) In: *Proceedings of RSS*.

Papadimitriou CH and Tsitsiklis JN (1987) The complexity of Markov decision processes. (d) *Mathematics of Operations Research* 12(3): 441–450.

Pineau J, Gordon G and Thrun S (2006) Anytime point-based approximations for large POMDPs. (d) *Journal of Artificial Intelligence Research* 27: 335–380.

Poupart P (2005) *Exploiting Structure to Efficiently Solve Large Scale Partially Observable Markov Decision Processes*. (d) Ph.D. thesis, University of Toronto.

Pynadath DV and Tambe M (2002) The communicative multiagent team decision problem: analyzing teamwork theories and models. (d) *Journal of Artificial Intelligence Research* 16: 389–423.

Rabinovich Z, Goldman CV and Rosenschein JS (2003) The complexity of multiagent systems: the price of silence. (d) In: *Proceedings of AAMAS*, pp. 1102–1103.

Roth M, Simmons R and Veloso M (2005) Decentralized communication strategies for coordinated multi-agent policies. (d) In: Schultz A, Parker L and Schneider F (eds.), *Multi-Robot Systems: From Swarms to Intelligent Automata*, volume IV. Dordrecht: Kluwer Academic Publishers, pp. 93–105.

Roy N, Gordon G and Thrun S (2005) Finding approximate POMDP solutions through belief compression. (d) *Journal of Artificial Intelligence Research* 23: 1–40.

Seuken S and Zilberstein S (2008) Formal models and algorithms for decentralized decision making under uncertainty. (d) *Autonomous Agents and Multi-Agent Systems* 17(2): 190–250.

Shah JA, Conrad PR and Williams BC (2009) Fast distributed multi-agent plan execution with dynamic task assignment and scheduling. (d) In: *Proceedings ICAPS*, pp. 289–296.

Simmons R and Koenig S (1995) Probabilistic robot navigation in partially observable environments. (d) In: *Proceedings International Joint Conference on Artificial Intelligence*, volume 2, pp. 1080–1087.

Sleight J and Durfee EH (2012) A decision-theoretic characterization of organizational influences. (d) In: *Proceedings of AAMAS*, volume 1, pp. 323–330.

Sondik EJ (1971) *The Optimal Control of Partially Observable Markov Processes*. (d) Ph.D. thesis, Stanford University.

Spaan M, Gonçalves N and Sequeira J (2010a) Multirobot coordination by auctioning POMDPs. (d) In: *Proceedings of ICRA*, pp. 1446–1451.

Spaan MTJ, Oliehoek FA and Vlassis N (2008) Multiagent planning under uncertainty with stochastic communication delays. (d) In: *Proceedings of ICAPS*, pp. 338–345.

Spaan MTJ, Veiga TS and Lima PU (2010b) Active cooperative perception in network robot systems using POMDPs. (d) In: *Proceedings of International Conference on Intelligent Robots and Systems*, pp. 4800–4805.

Spaan MTJ and Vlassis N (2004) A point-based POMDP algorithm for robot planning. (d) In: *Proceedings of the IEEE International Conference on Robotics and Automation*, New Orleans, Louisiana, pp. 2399–2404.

Stroupe A and Balch T (2005) Value-based action selection for observation with robot teams using probabilistic techniques. (d) *Robotics and Autonomous Systems* 50: 85–97.

Theocharous G and Kaelbling LP (2003) Approximate planning in POMDPs with macro-actions. (d) In: *Advances in Neural Information Processing Systems*.

Thrun S, Burgard W and Fox D (2005) *Probabilistic Robotics*. (d) Cambrideg, MA: MIT Press.

van der Sluis L (ed.) (2011) *Future Generation – Smartgrid Research in the Netherlands*. (d) TU Delft Library.

Viguria A, Maza I and Ollero A (2008) S+T: an algorithm for distributed multirobot task allocation based on services for improving robot cooperation. (d) In: *Proceedings of ICRA*, pp. 3163–3168.

Wong EM, Bourgault F and Furukawa T (2005) Multi-vehicle Bayesian search for multiple lost targets. (d) In: *Proceedings of ICRA*, pp. 3169–3174.

Appendix A: Probabilistic models

This appendix details the transition, observation and reward models used for each of the example applications in this paper.

A.1. Case study 1: environmental monitoring

Given the positions of the m critical points to monitor, $L^1, \ldots, L^m$, the factored state of a UAV i consists of its position in the grid (see Figure 6), $l_i \in \{1, \ldots, 40\}$; the contamination level of each critical point j, $status^j \in \{none, low, high\}$; and a factor to specify whether each critical point j is classified, $isClass^j \in \{0, 1\}$:

$$s_i = (l_i, status^1, isClass^1, \ldots, status^m, isClass^m). \quad (13)$$

The transition probability $p(s'|a, s)$ can be factorized and computed for the different factors in the following manner.

A.1.1. UAV transition model A UAV can select a classifying action $a_i \in \{classCont, classNotCont\}$, in which case it does not move and l_i will not change. However, if $a_i \in \{north, west, east, south\}$, the UAV will move to a desired goal cell $l_{goal} = f_1(l_i, a_i)$. Also, there is a set of additional cells where it may end up due to obstacle avoidance or other issues $\mathcal{C} = f_2(l_i, a_i)$ (see Figure 16a). Thus, if a_i is any of the movement actions, the transition for the UAV position is the following:

$$p(l_i'|a_i, l_i) = \begin{cases} 0.792, & \text{if } l_i' = l_{goal} \\ 0.108, & \text{if } l_i' = l_i \\ 0.025, & \text{if } l_i' \in \mathcal{C} \\ 0, & \text{otherwise.} \end{cases} \quad (14)$$

A UAV can only classify a critical point when it is on top of it. Thus, $p(isClass'^j = 1|a, isClass^j) = 1$ if $\exists i \setminus a_i \in \{classCont, classNotCont\}$, and $l_i = L^j$. Otherwise, the critical point is declassified with the following model: $p(isClass'^j = 1|a, isClass^j = 0) = 0$; and $p(isClass'^j = 0|a, isClass^j = 1) = p_{des}$.

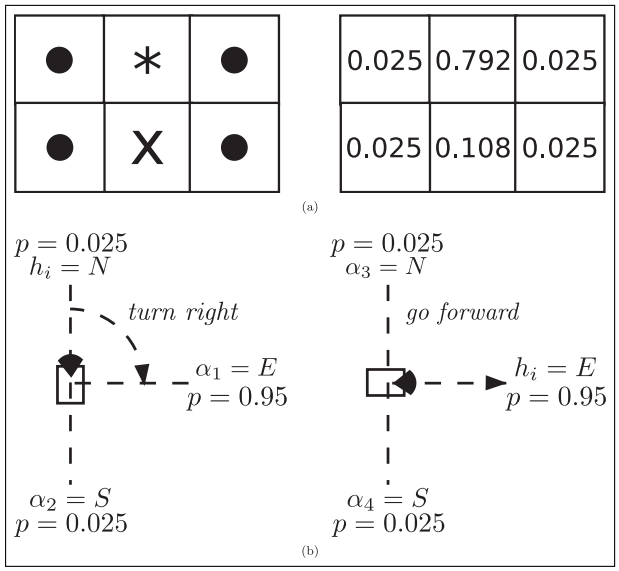

Fig. 16. Graphical examples of the transition models for the movement of the robots. (a) Position transition if the robot has to advance towards the north. On the left, the initial position is marked with a cross, the goal position with a star, and the cells in $\mathcal{C}$ with a dot. On the right, the transition probabilities. (b) Heading transition probabilities for two cases. On the left, the robot is pointing north and it is commanded to turn right. On the right, the robot pointing east and it is commanded to go forward.

Table 3. Transition model for the contamination level of an entry point j.

$status^j$	$status'^j$		
	none	*low*	*high*
none	0.9	0.1	0.0
low	0.1	0.8	0.1
high	0.0	0.1	0.9

A.1.2. Propagation model Regarding the contamination level, there is a propagation model. The critical points are connected in a directed graph and they can be of two types: entry points or normal points. Entry points have no parents and the contamination could appear or disappear on them independently. For instance, the critical point 1 in Figure 6 is an entry point. Their transition model is described in Table 3.

The points with one or more parents are normal points, and their levels of contamination depend on those from their parents. The contamination level is propagated from the parents to the children, so the level on a normal point will increase or decrease as the level on its parents does. Table 4 describes this propagation model for a point with a single parent. When there are more parents, effects of the different parents (Table 4) are combined.

A.1.3. Sensor model and reward The observations for each UAV, $z_i \in \{yes, no\}$, are conditionally independent,

Table 4. Transition model for the contamination level of a normal point j with a single parent k.

$status^j / status^k$	$status'^j$		
	none	*low*	*high*
none/none	1.0	0.0	0.0
none/low	0.9	0.1	0.0
none/high	0.85	0.15	0.0
low/none	0.1	0.9	0.0
low/low	0.0	1.0	0.0
low/high	0.0	0.9	0.1
high/none	0.0	0.15	0.85
high/low	0.0	0.1	0.9
high/high	0.0	0.0	1.0

so the observation probability function can be factorized $p(z|a,s') = \prod_{i=1}^{n} p(z_i|l'_i, status'^1, \ldots, status'^m)$. If $l'_i \notin \{L^1, \ldots, L^m\}$, then $p(z_i = yes|s'_i) = 0$. Otherwise,

$$p(z_i = yes|s'_i) = \begin{cases} 0.05, & \text{if } status'^j = none \\ 0.6, & \text{if } status'^j = low \\ 0.9, & \text{if } status'^j = high \end{cases} \quad (15)$$

where j is such that $l'_i = L^j$.

Finally, there is a local reward for each UAV i and behavior k, $R_i^k(s_i, a_i)$. The reward is only given when the UAV classifies a critical point. Therefore, there is no reward if $l_i \notin \{L^1, \ldots, L^m\}$ or $a_i \notin \{classCont, classNotCont\}$. Otherwise,

$$R_i^k(s_i, a_i) = \begin{cases} 10, & \text{if } a_i = classCont, \\ & \text{and } status^k \in \{low, high\} \\ -90, & \text{if } a_i = classCont, \\ & \text{and } status^k = none \\ 10, & \text{if } a_i = classNotCont, \\ & \text{and } status^k = none \\ -90, & \text{if } a_i = classNotCont, \\ & \text{and } status^k \in \{low, high\}. \end{cases} \quad (16)$$

Of course, if the critical point was already classified, there is no reward. Otherwise, the UAVs would keep classifying all of the time to obtain rewards and the policy computation would converge very slowly. Moreover, a small cost of 0.1 is assigned for the movement actions, whereas no cost is assigned for *stay*.

For the experiments with the MPOMDP model there is a joint reward function $R(s,a)$. On the one hand, if a UAV i is not classifying (i.e. $l_i \notin \{L^1, \ldots, L^m\}$ or $a_i \notin \{classCont, classNotCont\}$), it does not contribute to this joint reward. On the other hand, for each UAV i classifying, the value of the reward $R_i^k(s_i, a_i)$ is added to $R(s,a)$, being k such that $l_i = L^k$. However, if several UAVs are classifying the same critical point as *contaminated* or as *non-contaminated*, the local reward term is added only once. This is not to reward several times UAVs classifying the same point. Moreover, if there are different UAVs classifying the same critical point as *contaminated* and as *non-contaminated* at the same time, a reward value of -90 is added to $R(s,a)$ as penalization (instead of any local term $R_i^k(s_i, a_i)$).

A.2. Case study 2: cooperative tracking

The factored state of each robot i consists of its position in the grid (see Figure 9b), $l_i \in \{1, \ldots, 82\}$; its heading, $h_i \in \{north, west, south, east\}$; and the position of the target, $l_t \in \{1, \ldots, 82\}$: $s_i = (l_i, h_i, l_t)$. The local action for each robot is $a_i \in \{stay, turn\ right, turn\ left, go\ forward\}$. Moreover, the transition probability $p(s'|a,s)$ can be factorized into different components.

The target only moves to eight-connected cells at each iteration. Therefore, if n_8 is the number of eight-connected cells for a initial position l_t,

$$p(l'_t|l_t) = \begin{cases} 1/n_8, & \text{if } ||l'_t - l_t|| < 2 \\ 0, & \text{otherwise} \end{cases} \quad (17)$$

where the operator $||\cdot||$ represents Cartesian distance in the grid.

The action *stay* does not vary the position l_i nor the heading h_i of the robot. If $a_i \in \{turn\ right, turn\ left\}$, l_i does not vary, but the goal heading can be obtained by rotating 90° the initial one clockwise or counter-clockwise, respectively: $\alpha_1 = rot_{cw}(h_i)$ or $\alpha_1 = rot_{ccw}(h_i)$. If the robot rotates too much, it will end up in the next heading $\alpha_2 = rot_{cw}(\alpha_1)$ or $\alpha_2 = rot_{ccw}(\alpha_1)$, respectively (see Figure 16b for a graphical example):

$$p(h'_i|a_i, h_i) = \begin{cases} 0.025, & \text{if } h'_i = h_i \\ 0.95, & \text{if } h'_i = \alpha_1 \\ 0.025, & \text{if } h'_i = \alpha_2 \\ 0, & \text{otherwise.} \end{cases} \quad (18)$$

If the action is *go forward*, the robot heading can also vary due to an erroneous movement or an obstacle avoidance manoeuvre:

$$p(h'_i|a_i, h_i) = \begin{cases} 0.025, & \text{if } h'_i = \alpha_3 \\ 0.95, & \text{if } h'_i = h_i \\ 0.025, & \text{if } h'_i = \alpha_4 \\ 0, & \text{otherwise} \end{cases} \quad (19)$$

where $\alpha_3 = rot_{ccw}(h_i)$ and $\alpha_4 = rot_{cw}(h_i)$.

In addition, for each *go forward* action, there is a goal position (in front of the robot) $l_{goal} = f'_1(l_i, a_i)$, and a set of additional positions where the robot may end up due to obstacle avoidance or other issues $\mathcal{C} = f'_2(l_i, a_i)$. This model is similar to that in the previous case study (see Figure 16 for an example):

$$p(l'_i|a_i, l_i) = \begin{cases} 0.792, & \text{if } l'_i = l_{goal} \\ 0.108, & \text{if } l'_i = l_i \\ 0.025, & \text{if } l'_i \in \mathcal{C} \\ 0, & \text{otherwise.} \end{cases} \quad (20)$$

The observations for each robot, $z_i \in \{detected, non\ detected\}$, are conditionally independent, so the observation probability function can be factorized $p(z|a, s') = \prod_{i=1}^{n} p(z_i|l'_i, h'_i, l'_t)$. The target can only be detected if it is in the FOV of the robot. This FOV consists of a 3×4 rectangle in front of the robot (see Figure 9b).

$$p(z_i = detected|s'_i) = \begin{cases} p_D, & \text{if } l'_t \in FOV(l'_i, h'_i) \\ 0, & \text{otherwise.} \end{cases} \quad (21)$$

Finally, there is a local reward for each robot i and behavior k, $R_i^k(s_i, a_i)$. The reward is only given when the robot is tracking the target from a certain heading k, which depends on the behavior. Tracking the target means that it is located in one of the three cells in front of the robot (see Figure 9b):

$$R_i^k(s_i, a_i) = \begin{cases} 100, & \text{if } l_t \in \mathcal{D} \\ & \text{and } h_i = k \\ 0, & \text{otherwise} \end{cases} \quad (22)$$

where $\mathcal{D} \subset FOV(l_i, h_i)$, and $l_t \in \mathcal{D} \iff l_t \in FOV(l_i, h_i)$ and $||l_t - l_i|| < 2$.

Appendix B: Index to Multimedia Extensions

The multimedia extension page is found at http://www.ijrr.org

Table of Multimedia Extensions

Extension	Type	Description
1	Video	Experiments on cooperative tracking with multiple robots