Multi-Robot Coordinated Decision Making under Mixed Observability through Decentralized Data Fusion

J. Capitán, L. Merino and A. Ollero

***Abstract*— *Partially Observable Markov Decision Processes* (POMDPs) provide a sound mathematical framework to deal with robotic planning when tasks outcomes and perception are uncertain, but their main problem is scalability. This paper deals with planning in multi-robot teams, where this problem is even more evident. The paper presents a novel combination of two methods to cope with this complexity. *Mixed observability* can lead to simpler representations of the problem. The basic idea is to assume that some of the components in the state are fully observable, and solve the POMDP only on the partially observable part. For multi-robot teams, the complexity can be also reduced by using a *decentralized* approach in which robots have no knowledge about others' actions. In this sense, an implicit coordination will be derived from the sharing of the beliefs among the robots. The paper illustrates the advantages of the methods proposed in a multi-robot tracking application. Simulations and actual experiments are shown.**

I. INTRODUCTION

Techniques for planning under uncertainty are being applied more and more to robotics (see for example [1], [2]). In all cases, the underlying idea is that the state is only partially observable, and thus, the planning objective is to find a policy that indicates which action a robot should take given the information available (the history of actions and observations gathered by the robot so far, called the information space) in order to reach a goal. This paper is concerned with decision making in applications that consider teams of networked robots.

Considering probabilistic models and a Markovian environment, a belief state can be used to represent the information space; in this case, POMDP techniques provide an elegant way to model the interaction of a robot with its environment. Based on prior knowledge of the sensor's model and the environment dynamics, policies which indicate the robot how to act given the belief can be computed. These policies can be extracted by optimizing iteratively a certain value function over the belief space. However, the main problem of POMDPs is scalability. The optimal plan has to be searched on the belief space, which can be very large. This drawback is even more evident in multi-robot teams, as the space of actions and observations increases exponentially with the number of robots.

Approximate point-based methods to obtain POMDP policies for large state spaces have been studied ([3], [4], [5]). They restrict the optimization procedure to a bounded set of feasible sampled beliefs. Particularly, [4] proposes a point-based solver called Perseus, where no computation is needed for all the belief points at every iteration, hence improving performance. In [6], an extension of Perseus is presented. It is called *Symbolic* Perseus and uses Algebraic Decision Diagrams (ADDs) [7] in order to optimize the operations for factored POMDPs. Besides, [8] proposes SARSOP, which maintains a tree-shaped set of reachable beliefs that is expanded at every iteration using the best policy so far.

Other authors try to cope with the high dimensionality of the belief space by exploiting the idea that many robotic systems often have mixed observability, i.e. although the state is not fully observable, some components might be. Thus, [9] propose MOMDPs (*Mixed Observability Markov Decision Processes*), which separate the fully and partially observable components, leading to a lower-dimensional representation of the belief space.

Nevertheless, these techniques face ultimately an scalability problem with the number of robots in a team. A decentralized (Dec-)POMDP [10] is a suitable model for multi-agent planning considering decentralized execution. Basically, in a Dec-POMDP each agent only access to local information, and no communication is available. Thus, the agents reason on all the potential action-observation histories of the other agents (which cannot be observed) in order to compute an optimal policy. Despite their decentralized execution, Dec-POMDP policies are computed in a centralized manner, which is a NEXP-complete problem [10].

The idea presented in this paper is that communication and data fusion between the robots can induce a common Markovian belief signal, which can be used to coordinate the robots' local plans, alleviating the mentioned complexity and scaling with the number of robots. Hence, MOMDPs are considered for local planning, whereas Decentralized data fusion (DDF) is used to induce a common belief signal among the robots and coordinate the plan execution.

The idea is similar to that proposed in [11], [12], where a distinction between cooperative and coordinated information-theoretic approaches is made, proposing the latter to control fleets of robots. In a coordinated approach, the members of the team have no knowledge about the others' models or control actions, but they exchange some information that may influence implicitly other members' subsequent decisions.

This work was partially supported by the European Commission under the Cooperating Objects NETwork of Excellence CONET (contract FP7-ICT-224053) and the Spanish project ROBAIR (DPI2008-03847)

L. Merino is with Pablo de Olavide University, Carretera Utrera km1, 41013, Sevilla (Spain). lmercab@upo.es

J. Capitán and A. Ollero are with the University of Seville, Escuela Superior de Ingenieros, Camino de los Descubrimientos s/n, 41092 Sevilla (Spain). {jescap,aollero}@cartuja.us.es

A. Ollero is also with the Center for Advanced Aerospace Technologies (CATEC), Parque Tecnológico y Aeronáutico de Andalucía, C. Wilbur y Orville Wright 17-19-21, 41309, La Rinconada (Spain)

Hence, sharing fused perception information or the impact of others' control actions over a certain objective function, and acting locally, a coordinated behavior can be obtained. However, these approaches consider mainly greedy control algorithms that try to obtain the next suitable action and do not reason on long-term goals.

The paper is structured as follows: Sections II and III give some theoretical background about POMDPs and MOMDPs; Section IV describes the communication and data fusion process to coordinate the robots; Section V details a coordinated tracking application for multiple robots to illustrate the ideas; and Sections VI and VII explain the experimental results and conclusions respectively.

II. POMDP Model

Formally, a POMDP is defined by the tuple $\langle S, A, Z, T, O, R, h, \gamma \rangle$. 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 made and a reward 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 total expected reward 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 to actions in the form $\pi(b) \longrightarrow a$, so that the total expected reward is maximized. This expected reward gathered by following π starting from belief b is called the value function:

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

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: $\pi^*(b) = \arg\max_{\pi} V^{\pi}(b)$.

III. MOMDP Model

Even for a finite set of $|S|$ states, π is defined over a $(|S| - 1)$-dimensional continuous belief space. The key in MOMDPs is to gain computational efficiency by solving a number of lower-dimensional POMDPs instead of the original one. Thus, all operations work on lower-dimensional belief spaces, which can lead to a relevant improvement in the performance.

The MOMDP [9] is represented as a factored POMDP where the state vector is composed of two different parts. Component x is the fully observable part of the original state s and y is another vector representing the partially observable part. Thus, the state is specified by $s = (x, y)$, and the state space is $S = X \times Y$, where X is the set with all possible values for x and Y all possible values for y.

Formally, the MOMDP is defined by the tuple $\langle X, Y, A, Z, T_x, T_y, O, R, h, \gamma \rangle$. The components are the same as in the POMDP case, but the transition function T is now decomposed into T_x and T_y. $T_x(x', a, x, y) = p(x'|a, x, y)$ gives the probability that fully observable state component has value x' if the robot takes action a in state (x, y). $T_y(y', x', a, x, y) = p(y'|x', a, x, y)$ gives the probability that the partially observable state component has value y' if the robot takes action a in state (x, y) and the fully observable state component has value x'.

In a MOMDP, since it can be observed, there is no need to maintain a belief over x. Therefore, it can be excluded in order to just maintain a belief $b_y(y)$, which is a probability distribution over y. Any belief $b \in B$ on the complete system state $s = (x, y)$ is then represented as (x, b_y), where $b_y \in B_y$. Furthermore, for each value x of the fully observable state component, a belief space $B_y(x) = \{(x, b_y) | b_y \in B_y\}$ is associated. Here, every $B_y(x)$ is a subspace in B, and B is the union of these subspaces $B = \bigcup_{x \in X} B_y(x)$.

Note that while B has $|X||Y|$ dimensions, each $B_y(x)$ has only $|Y|$ dimensions. Therefore, the objective of representing a high-dimensional space as a union of lower-dimensional subspaces is achieved. Moreover, when $|Y|$ is small, a remarkable computational improvement can be reached, since operations to solve the MOMDP are performed over the subspaces $B_y(x)$.

Now, since every belief is represented by (x, b_y), [9] prove that the value function can be rewritten as:

$$V(x, b_y) = \max_{\alpha \in \Gamma_y(x)} \alpha \cdot b_y \tag{3}$$

where for each x, $\Gamma_y(x)$ is a set of α-vectors defined over $B_y(x)$. Therefore, the value function is now a collection of sets of $|Y|$-dimensional vectors, and the value of the observable component x determines which $\Gamma_y(x)$ is selected. Then, the maximum over $\Gamma_y(x)$ is calculated. Now, since the vectors have $|Y|$ dimensions instead of $|X||Y|$, the execution of the policy is also faster for a MOMDP.

A. Point-Based MOMDP Solvers

Once MOMDPs have been introduced, the question is how to solve them. Fortunately, with some modifications, the same point-based algorithms for POMDPs are valid now. Since the value function consists of a set of α-vectors for every $x \in X$, the idea is to run an independent value iteration

for each of them separately. In [9], for instance, the point-based solver SARSOP for POMDPs is modified in order to cope with MOMDPs. Here, Symbolic Perseus [6] has been also extended to cope with mixed observability policies. The resulting algorithm has been named MO-Symbolic Perseus and it allows a useful comparison with SARSOP for MOMDP users. Moreover, recalling that Symbolic Perseus exploits the factored representation of POMDPs and the ADD structures in order to optimize the original Perseus, MO-Symbolic Perseus can be of great interest in certain domains where the variables are not very coupled.

The two main steps to adapt point-based POMDP solvers to deal with MOMDPs are the belief update and the backup operation. In a POMDP, the belief update was determined by (1). However, since $b_y(y) = b(s)$ when $s = (x, y)$ in a MOMDP, the equation can be transformed:

$$\begin{aligned} b'_y(y') =& \eta O(z, a, x', y') \\ & \sum_{y \in Y} T_y(y', x', a, x, y) T_x(x', a, x, y) b_y(y) \end{aligned} \tag{4}$$

When the belief is updated, the values of the observable states before (x) and after (x') taking an action are known. Hence, (4) is particularized for specific values of those variables.

The other major modification in MOMDP solvers is the backup operation. Based on all the considerations made for MOMDPs, [9] show how the backup operation included in the original Perseus can be rewritten for MOMDPs:

$$\texttt{backup}(b_y) = \arg\max_{a \in A} \; b_y \cdot \alpha_a, \text{ where} \tag{5}$$

$$\begin{aligned} \alpha_a(y) =& R^x_a + \gamma \sum_{z \in Z} \sum_{x' \in X} \sum_{y' \in Y} (T_x(x', a, x, y) \\ & T_y(y', x', a, x, y) O(z, a, x', y') \alpha_{a,x',z}(y')) \end{aligned} \tag{6}$$

Finally, note that:

$$\alpha_{a,x',z}(y') = \arg\max_{\alpha \in \Gamma_{y,n-1}(x')} \; \alpha(y') \cdot b^z_{a,y}(y') \tag{7}$$

In this case, unlike (4), all the possible x' must be taken into account. Even though x' and z are concrete values, they cannot be known beforehand like x, so all their values must be considered and weighted by their probabilities.

Apart from the steps mentioned above, MO-Symbolic Perseus also needs some modifications when initializing. First, a different set of beliefs $B_y(x)$ over the component y must be sampled for every value of x. In case y is probabilistically independent of x, the same set of beliefs over y could be used for every x. Moreover, all the value functions $\Gamma_y(x)$ are initialized separately with the same values as in the original Symbolic Perseus.

IV. Coordination through Decentralized Data Fusion

Given a multi-robot planning problem, the first option is to solve the above MOMDP for the whole team, considering all the potential joint actions and observations, but this approach ultimately does not scale with the number of robots. Hence, for a team of N robots, a decentralized scheme is proposed, where each robot i solves its own MOMDP without considering the other robots actions. However, if the robots solve independent MOMDPs, and use only their local information (action and observations), no coordination or cooperation is achieved. On the other hand, if communication is allowed among the robots and the same belief state can be recovered locally, this would represent a coordination signal that summarizes all the information gathered by the fleet. This way, the execution of the policies of the different robots will be coordinated. The solution should be of lower quality than a fully cooperative centralized solution, but it can represent a tradeoff between quality and complexity.

Then, once the policies have been calculated, the coordination is achieved during the execution phase by sharing a common belief state $b_{cen}(y)$ that considers information from all the robots. If $a^J = \langle a^1, \cdots, a^N \rangle$ is the joint action and $z^J = \langle z^1, \cdots, z^N \rangle$ the joint measurement, a centralized node with access to all the information would update the belief according to (4):

$$b'_{cen}(y') = \eta p(z^J | a^J, x', y') \sum_{y \in Y} p(x', y' | a^J, x, y) b_{cen}(y) \tag{8}$$

The question is how to recover this centralized belief in a decentralized manner. Assuming that the data gathered by the different robots at any time instant are *conditionally independent* given the state at that instant s', i.e. $p(z^J | a^J, x', y') = \prod_i p(z^i | a^i, x', y')$, and the prediction does not depend on the robot actions (or the robot actions are known when predicting), it is possible to combine locally the received belief from other robots with the local one of robot i, $b'_i(y')$, to recover the centralized belief [13], [14]:

$$b'_{cen}(y') \propto b'_i(y') \prod_{j \neq i} \frac{b'_j(y')}{b'_{ij}(y')} \tag{9}$$

where $b'_{ij}(y')$ represents the common information between the robots i and j (i.e. information previously exchanged between the robots). This common information can be maintained by a separate filter called channel filter [15]. If there are loops in the information channels, the problem of double-counting should be taken into account as well. The authors have shown previously [14] that it is possible, by including delayed states in the belief, to obtain locally the same belief as in a centralized node with access to all the information available.

V. MOMDP for Target Tracking

In order to illustrate the proposed approach, an application for tracking a target by means of multiple robots is

considered here. In this problem there is a moving target and a team of N robots which are the pursuers. Each of these robots carries a sensor which determines whether the target is visible or not within its field of view (FOV). Then, the objective is to find the target in the environment and localize it as well as possible.

The state is composed of the position of the target and the position and heading (*north*, *west*, *south* or *east*) of the pursuer robots. The state space is discretized into a cell grid, and a map of the scenario is assumed to be known. At each time step, each robot can choose among four possible actions: *stay*, *turn right*, *turn left* or *go forward*. *stay* means doing nothing; when *turning*, the robot changes its heading 90^o degrees; and when *going forward*, it moves to the cell ahead. Nonetheless, noisy transition functions for the states of the robots are considered. Besides, 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 8-connected cells with the same probability (only non-obstacle cells are considered in order to calculate that probability).

In addition, every sensor provides a boolean measurement: *detected* or *non-detected*. These sensors proceed as it follows, if the target is out of its 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 are heterogeneous in the sense that the sensor's FOV and p_D could vary from one robot to another.

Finally, the design of the reward function so that the target is tracked by the team of robots is crucial. Since some cooperation is desirable within the heterogeneous team, a different behavior is assigned to each robot. Thus, the reward function also depends on the specific robot: $\{R^1(x, y, a), R^2(x, y, a), \cdots, R^N(x, y, a)\}$. For all the members of the team, no cost is assigned to the action *stay*, whereas a cost of 1 is associated to the other actions.

This application is a fair example of how MOMDPs can help to reduce the belief space dimensionality. Here, even though robots and target locations are considered within the state, the one involving a greater uncertainty is the latter. Here, the locations of the robots are assumed to be observable (they could be obtained accurately enough by means of the on-board sensors and the available map, at least for the cell resolution), remaining as non-observable the target's position. Thus, the non-observable part of the state consists of the target location, $y = t_l$, whereas the fully observable part consists of all the robots locations and headings, $x = (r_l^1, r_h^1, \cdots, r_l^N, r_h^N)$. In this case, for a 10x10-cell grid, there would be 400 possible states for each pursuer and 100 for the target, which means, for a single robot, to reduce a POMDP with a 40,000-dimensional belief space to a union of 400 disjoint 100-dimensional subspaces.

Then, the proposed coordinated approach will be applied. That means that each robot i solves its own MOMDP without considering the other robots. That MOMDP has states $x^i = (r_l^i, r_h^i)$ and $y^i = t_l$, and reward $R^i(x, y, a)$, being possible to specify a different strategy for each robot.

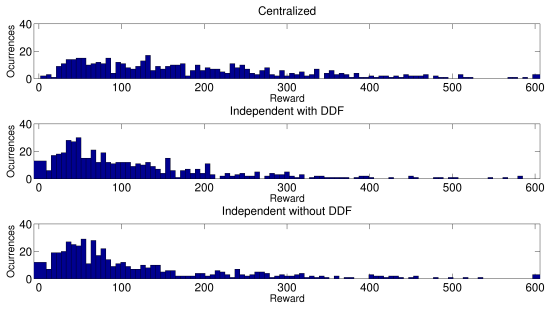

Fig. 1: Histograms of the average rewards obtained during 500 simulations for the different approaches.

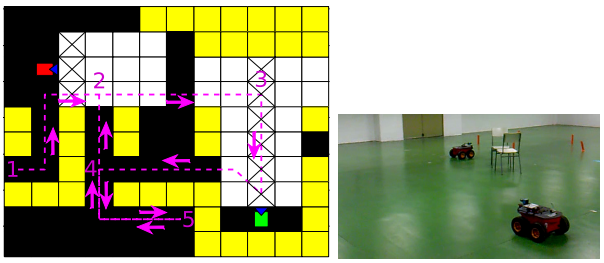

Fig. 2: a) Simulated environment and field of view (white cells) for each robot. If the target is in one of the cells with crosses, a high reward is obtained. The path of the target is also shown. b) Image of the experiments in the real CONET testbed.

With this decentralized approach based on MOMDPs, coordination arises implicitly due to the fused belief, and there is no need to solve a MOMDP for the whole team nor a POMDP, whose complexities grow exponentially with N. Of course, the policy is not optimal, as the robots do not reason about the other robot actions, but, as it will be seen, through proper design of the rewards and the communication between robots, it can be obtained a helpful coordinated behavior for many applications.

VI. Experiments

A. Simple scenario

A simple target tracking scenario was simulated in order to highlight the differences between the proposed approach and a fully cooperative centralized solution. The model is the one explained above particularized for a map with 11 available cells. Then, two pursuer robots are considered with equal FOVs, composed by a single cell in front of them and $p_D = 0.9$. Each robot gets a high reward $(+100)$ when the target is in the cell of its FOV, otherwise the reward is zero.

A MOMDP for a single robot without considering the other and a fully cooperative MOMDP considering both robots were solved. The reward for the latter model was simply the sum of the independent rewards for each robot. Moreover, both policies were calculated with SARSOP in a computer with an Intel Core 2 Duo processor @2.47GHz and 2.9GB, being the former computed for 0.2 seconds and the latter for 4677. Then, three different approaches were tested:

	Coordinated	Not coordinated
MO-Symbolic Perseus	246.13 ± 3.33	209.38 ± 3.17
SARSOP	279.64 ± 4.67	197.76 ± 3.10

TABLE I: Rewards (with 95% confidence interval) obtained with and without coordination (DDF) during the simulations.

(i) the fully cooperative policy; (ii) independent policies for each robot but using DDF; (iii) independent policies for each robot without sharing any information. Fig. 1 depicts histograms for the average rewards obtained during 500 simulations of 150 steps each one.

In this case, with much less computation time, the quality of the proposed coordinated approach was quite close to the fully cooperative. Actually, computing the single-robot policy for 20 seconds the proposed approach outperformed the centralized. In general, the reward is usually higher when sharing information compared to no data fusion. Nonetheless, due to the small size of the map, differences were not very substantial in this example.

B. Simulations

Additional simulations for more complex environments were conducted in order to show the advantages of introducing *mixed observability* and coordination through DDF. A simulation environment was created from the original testbed of the Cooperating Objects Network of Excellence (CONET)[1]. The map of the real testbed was discretized into 2x2-meter cells and resulted in the occupancy grid of 12x10 dimensions shown in Fig. 2, where cells representing obstacles are in yellow.

Again, a team with two robots is considered. Nevertheless, their perception capabilities are different. The first robot carries a more accurate sensor ($p_D = 0.9$) but its FOV is smaller, whereas the second has a wider FOV but is less accurate ($p_D = 0.8$) (see Fig. 2). That is reasonable, because many vision-based detectors work more accurately when the FOV of the scene is more restricted. Therefore, the role of the robot with the wider FOV would be to survey a big area from a distant position whereas the robot with the more accurate sensor would try to get closer to confirm the target detection. Hence, the first robot gets a high reward (+100) when the target is in one of the closest cells of its FOV, but the second robot gets a high reward (+100) when the target is in one of the central cells of its FOV (see Fig. 2); otherwise the reward is zero.

First, the decentralized approach with independent MOMDPs proposed in Section V was tested for this scenario. Independent policies were calculated for each robot with SARSOP and MO-Symbolic Perseus. Then, 500 simulations of 150 steps each one were performed with the robots starting at random positions and the target following the fixed path depicted in Fig. 2. In order to include some uncertainty in its behavior, at every time step, the target could (with equal probability) either stay in the same cell or follow the path.

[1]http://www.cooperating-objects.org

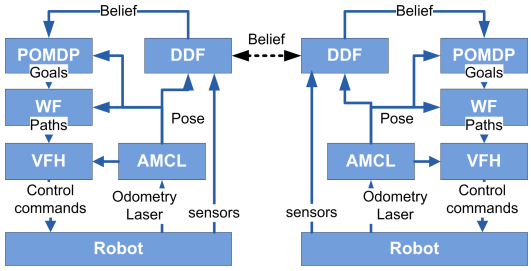

Fig. 3: Software architecture of the real robots.

Map size	Model	Time (s)	Precision	Reward
12 cells	MOMDP	0.18	21.97	125.87 ± 6.19
	POMDP	1517	21.99	88.74 ± 8.90
80 cells	MOMDP	1054	54.89	64.12 ± 8.61
	POMDP	3169	54.80	44.49 ± 8.70
120 cells	MOMDP	3071	57.42	42.58 ± 5.84
	POMDP	3081	47.30	40.24 ± 7.18

TABLE II: Evaluation of POMDP and MOMDP policies for a single robot (rewards with their 95% confidence interval).

Table I shows the average rewards obtained with and without coordination (DDF). It can be seen how, for both solvers, the use of DDF produces an increase of the average reward. It is also crucial to remark that a comparison with the fully coordinated MOMDP is not included here due to the complexity of the domain. Both solvers run out of memory (with the same computer mentioned in the previous section) trying to compute a fully coordinated policy.

Finally, to show the advantage of introducing the *mixed observability* in the model, some simulations (100 runs of 150 steps) with a single pursuer robot were carried out varying the size of the scenario and comparing with a standard POMDP solution. The results of the average rewards for both, POMDP and MOMDP policies, are summarized in Table II. All the policies were calculated with the same solver (SARSOP) and the same computer (the one mentioned above), and their precision measured, this meaning the distance between the upper and lower bounds of the value function [8]. For the smallest and medium scenarios, it can be seen that the MOMDP policy can achieve better average rewards with much less computation time. The biggest scenario tries to show how the results are better for the MOMDP after similar computation time.

C. Real experiments

In order to show the applicability of the approach, some real experiments were conducted. In these experiments, two robots were used to follow a target represented by a third robot (see Fig. 2). The models and scenario considered were the same as in the simulation described above.

Figure 3 shows the software modules used by the robots. Player [16] was used to control the robots, which were able to localize themselves within the map. Besides, a path planning algorithm was used to obtain the path to the high level goals provided by the MOMDP controller (next cell to move), whereas a local navigation algorithm was used to safely

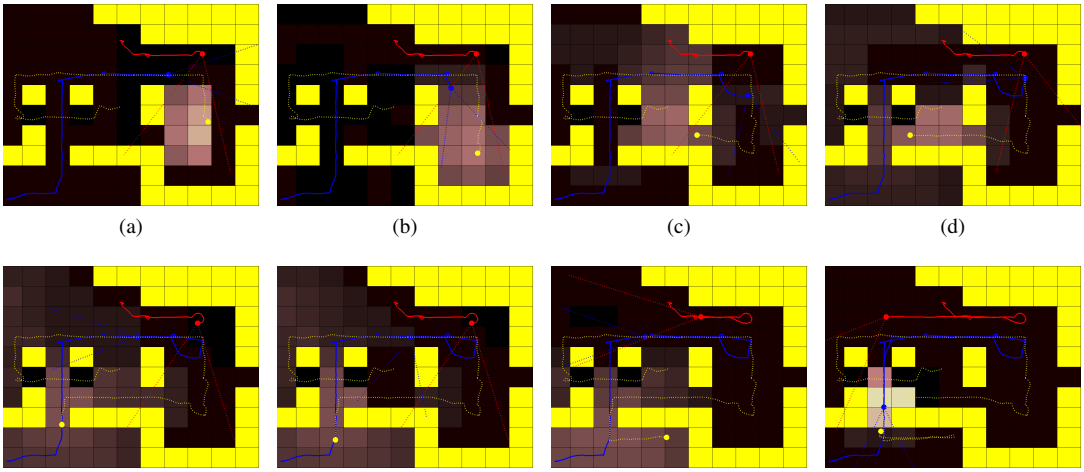

Fig. 4: Screenshots of a real experiment using coordinated robots. The color scale of the cells represents the (fused) belief from the blue robot. In yellow the target. The FOVs of the robots are also represented. Yellow cells cannot be reached.

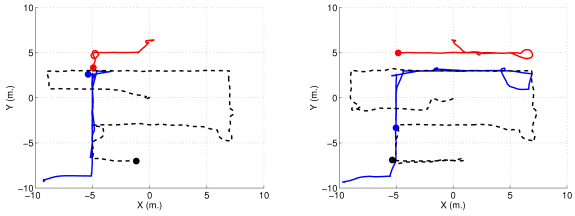

Fig. 5: Paths of the target (black) and robots. Left: no coordination. Right: coordination. In the first case the robots lost the target in part of the scenario.

navigate the given path. Each robot had an estimation filter implementing the DDF scheme of Section IV and a MOMDP controller that executed the obtained policies.

Results of the experiment with coordinated execution[2] are summarized in Fig. 4. The localization of the target is improved, as it can be seen in screenshot 4a, where the belief state of the blue robot is narrow even though the target is not in its FOV. Moreover, a certain coordination arises between the robots. For instance, in screenshots 4c, 4d, 4e, the target escapes and the blue robot goes through its lowest cost path while the red robot keeps observing the place, as there is a certain probability of the target coming back and it has a larger FOV. When most of the probability mass is in the lower left room (4f), both robots eventually move to that direction (4g), and the blue robot moves inside the room while the red awaits observing from afar (4h). Fig. 5 compares the trajectories of the robots and target in two different experiments, with and without coordination.

VII. Conclusions

The scalability of POMDP models is a concern for their application to multi-robot planning. This paper addresses decentralized data fusion as a manner to coordinate independent plans computed by local MOMDPs. Since each robot does not reason about the others' actions, this approach is scalable. The main drawback is that sub-optimal policies are obtained, and no intentional cooperation is considered. However, a proper design of the local rewards allow to cope with different applications with a coordinated team. Thus, the paper mainly contributes showing through simulations the advantages of introducing MOMDPs and a coordinated decentralized solution versus a fully cooperative solution. Real experiments are used to demonstrate the applicability of the approach. Moreover, a modification of the algorithm Symbolic Perseus to deal with mixed observability is proposed and compared.

Future work includes demonstration with fleets of more robots. A guideline to design the reward functions to achieve a desired coordination would be of interest as well. Certainly, better plans could be obtained if the actions of several robots were considered during the planning phase. Thus, another direction is to include other robots' actions in the planning phase, but trying to maintain locality (that is, to consider only the actions of potential neighbor robots, no the whole fleet). This should lead to better policies, but maintaining the scalability of the approach.

[2]See http://grvc.us.es/staff/jescap/expCONET3.mp4

References

[1] N. Vlassis, G. Gordon, and J. Pineau, "Planning under uncertainty in robotics," *Robotics and Autonomous Systems*, vol. 54, no. 11, 2006, special issue.

[2] S. Prentice and N. Roy, "The Belief Roadmap: Efficient Planning in Belief Space by Factoring the Covariance," *The International Journal of Robotics Research*, vol. 28, no. 11-12, pp. 1448–1465, November/December 2009.

[3] J. Pineau, G. Gordon, and S. Thrun, "Point-based value iteration: An anytime algorithm for POMDPs," in *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*, 2003, pp. 1025–1032.

[4] M. T. J. Spaan and N. Vlassis, "Perseus: Randomized point-based value iteration for POMDPs," *Journal of Artificial Intelligence Research*, vol. 24, pp. 195–220, 2005.

[5] T. Smith and R. Simmons, "Point-based POMDP algorithms: improved analysis and implementation," in *Proceedings of the 21th Conference on Uncertainty in Artificial Intelligence*, 2005, pp. 542–547.

[6] P. Poupart, "Exploiting Structure to Efficiently Solve Large Scale Partially Observable Markov Decision Processes," Ph.D. dissertation, University of Toronto, 2005.

[7] J. Hoey, R. St-Aubin, A. Hu, and C. Boutilier, "SPUDD: Stochastic planning using decision diagrams," in *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 1999.

[8] H. Kurniawati, D. Hsu, and W. Lee, "SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces," in *Proc. Robotics: Science and Systems*, 2008.

[9] S. C. Ong, S. W. Png, D. Hsu, and W. S. Lee, "POMDPs for Robotic Tasks with Mixed Observability," in *Robotics: Science and Systems Conference*, 2009.

[10] F. A. Oliehoek, M. T. Spaan, and N. Vlassis, "Optimal and approximate q-value functions for decentralized POMDPs," *Journal of Artificial Intelligence Research*, 2008.

[11] B. Grocholsky, A. Makarenko, T. Kaupp, and H. F. Durrant-Whyte, *Lecture notes in Computer Science*. Springer, 2003, vol. 2634, ch. Scalable Control of Decentralised Sensor Platforms.

[12] G. Mathews, H. Durrant-Whyte, and M. Prokopenko, *Advances in Applied Self-Organizing Systems*. Springer-Verlag, 2008, ch. Decentralized Decision Making for Multiagent Systems.

[13] E. Nettleton, H. Durrant-Whyte, and S. Sukkarieh, "A robust architecture for decentralised data fusion," in *Proc. of the International Conference on Advanced Robotics (ICAR)*, 2003.

[14] J. Capitan, L. Merino, F. Caballero, and A. Ollero, "Delayed-State Information Filter for Cooperative Decentralized Tracking," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2009.

[15] F. Bourgault and H. Durrant-Whyte, "Communication in general decentralized filters and the coordinated search strategy," in *Proc. of The 7th Int. Conf. on Information Fusion*, 2004, pp. 723–730.

[16] B. Gerkey, R. Vaughan, and A. Howard, "The Player/Stage Project: Tools for Multi-Robot and Distributed Sensor Systems," in *Proceedings of the 11th International Conference on Advanced Robotics, ICAR*, 2003, pp. 317–323.