Coordinated Multi-Robot Exploration under Communication Constraints using Decentralized Markov Decision Processes

Laëtitia Matignon Laurent Jeanpierre
Abdel-Illah Mouaddib*
Université de Caen Basse-Normandie, CNRS - UMR6072 GREYC - F-14032 Caen, France

Abstract

Recent works on multi-agent sequential decision making using decentralized partially observable Markov decision processes have been concerned with interaction-oriented resolution techniques and provide promising results. These techniques take advantage of local interactions and coordination. In this paper, we propose an approach based on an interaction-oriented resolution of decentralized decision makers. To this end, distributed value functions (DVF) have been used by decoupling the multi-agent problem into a set of individual agent problems. However existing DVF techniques assume permanent and free communication between the agents. In this paper, we extend the DVF methodology to address full local observability, limited share of information and communication breaks. We apply our new DVF in a real-world application consisting of multi-robot exploration where each robot computes locally a strategy that minimizes the interactions between the robots and maximizes the space coverage of the team even under communication constraints. Our technique has been implemented and evaluated in simulation and in real-world scenarios during a robotic challenge for the exploration and mapping of an unknown environment. Experimental results from real-world scenarios and from the challenge are given where our system was vice-champion. in simulation and in real-world scenarios during a robotic challenge for the exploration and mapping of an unknown environment by mobile robots that could be seen as more general than Pentagone or ISR environments. Experimental results from real-world scenarios and from the challenge are given where our system was vice-champion.

*In sabbatical year at LAAS-CNRS.

Introduction

Recent advancements concerning the resolution of decision-theoretic models based on Decentralized Partially Observable Markov Decision Processes (Dec-POMDPs) allowed a notable increase in the size of the problems that have been solved. Especially, one of the directions that attracts more and more from attention this community is to take advantage of local interactions and coordination with an interaction-oriented (IO) resolution [7, 11, 18]. Such approaches relax the most restrictive and complex assumption consisting in considering that agents are permanently in interaction. They become a promising direction concerning real-world applications of decentralized decision makers.

The approach developed in this paper is primary motivated by using a Dec-POMDP solved with IO techniques for an exploration and mapping multi-robot system. This system has been developed and applied successfully in real-world scenarios during a DGA[1]/ANR[2] robotic challenge named CAROTTE for the exploration, mapping and object recognition by mobile robots. In this paper, the distributed SLAM aspect is out of the paper scope and we focus only on the decision model. We consider that robots are independent and can share limited information by communication leading to some kind of observability completion. However a particularly significant challenge is the communication as potential communication breaks can happen leading to a loss of information that are shared between robots.

This paper concerns the global and local coordination of decentralized decision makers under the assumptions of full local observability, limited share of information between the agents and breaks in communication. Global coordination consists in allocating appropriately goals for the individual robots and minimizing the interactions that lead to conflicts between the members of the team. Local coordination is the resolution of close interactions. The assumptions we made are not simultaneously addressed in the litterature. For instance existing IO resolutions of decision models do not consider together these hypotheses; classical negotiation based techniques assume permanent communication; and existing multi-robot exploration approaches that consider communication constraints only cope with limited communication range issue and do not address the problem of failures as stochastic breaks in communication. So we introduce in this paper a new IO resolution method for decentralized decision models that handles limited share of information and breaks in communication.

In the sequel, we present first related works about the interaction-oriented resolution of Dec-POMDPs and multi-robot exploration. This is followed by a background of our work. Second, we introduce the DVF approach and its extension to support communication breaks. Then the application of this model to multi-robot exploration is detailed. Finally, experiments from real robot scenarios and simulations are given to demonstrate our method effectiveness when communication is

[1] French Defense Procurement Agency.

[2] French National Research Agency.

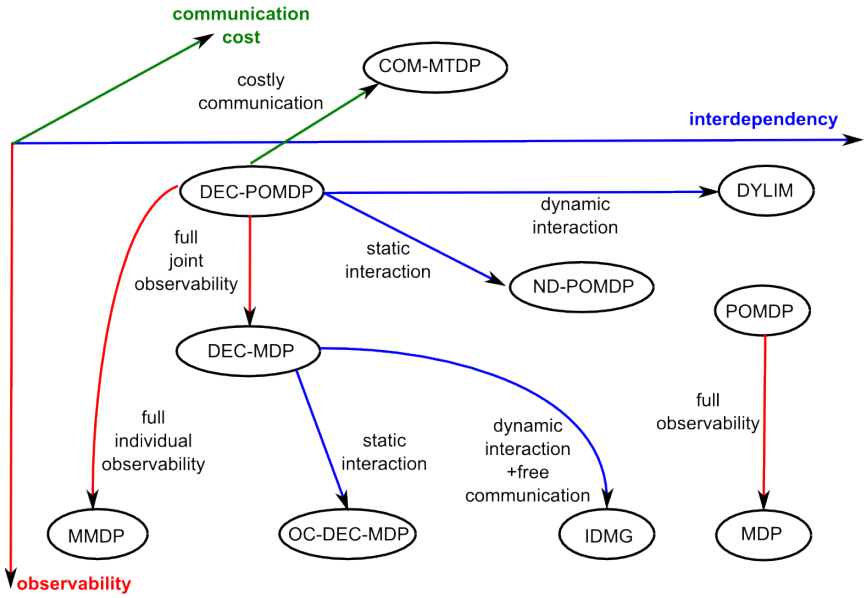


Figure 1: Decision models according to interdependency, communication and observability.

not reliable before concluding.

Related Work

Interaction-Oriented Models

Much of the work in multi-agent models address the Dec-POMDP complexity of resolution through one or more of the three directions: *observability* depending on whether each agent has complete or partial knowledge about the state of the world; *communication* according to the possibility and the cost of sharing information between the agents; and *interdependency* exploiting the structure of the problem such as locality of interaction, decomposition of rewards and independence between the agents. All existing Dec-POMDPs approaches use different assumptions about these directions. Fig. 1 summarizes some of these models and how they are related with parameters to consider to switch from one to another. Single-agent approaches (MDP and POMDP) depend on whether the agent's observability about the world is complete or partial.

As regards multi-agent models and the observability, the model moves from Dec-POMDP to Dec-MDP when the collective observability is complete and to MMDP when the individual observability is full. Recent promising works exploit the interdependency with an interaction-oriented (IO) resolution of Dec-POMDPs. Introducing some structures leads to new models which are based on a set of interactive individual decision making problems and thus reduces the complexity of

solving Dec-POMDPs. The ND-POMDP [12] model is a static interactions model approach, meaning that an agent is always interacting with the same subset of neighbors. In case of full collective observability and static graph of interactions, Dec-MDP moves to OC-Dec-MDP [3, 4]. Interacting all the time with the same agents is not realistic. Thus models have been proposed that use dynamic interactions such that each agent interacts with an evolving set of agents: IDMG [17] and DyLIM [7]. In all of these models, no explicit communications are assumed except for the IDMG model which has unlimited and free communication between agents interacting together. Few models study the cost of communication such as COM-MTDP [15].

Multi-Robot Exploration

Multi-robot exploration has received considerable attention in recent years. Various exploration strategies have been proposed that mainly differ by the way global coordination is achieved. In [6, 19], global coordination is centralized. The utility of each target is computed as a compromise between the gain expected at this target (expected area discovered when the target will be reached taking into account the possible overlap in between robot sensors) and the cost for reaching this target. Global coordination is accomplished by assigning different targets to the robots, thus maximising the coverage and reducing the overlap between explored areas of each robot. Global coordination can also be decentralized as in [21] where robots bid on targets to negotiate their assignments. Most approaches assume that robots maintain constant communication while exploring to share the information they gathered and their locations. However, permanent and free communication is seldom the case in practice and a significant challenge is to account for potential communication drop-out and failures. Some recent approaches consider the constraint of a limited communication range. Burgard et al. [6] apply their multi-robot strategy to each sub-team of robots which are able to communicate with each other. This leads in the worst case to a situation in which all robots individually explore the whole environment. Powers, Balch, and Lab [13] try to maintain a team connectivity during the exploration for the robots to remain in constant communication. Hoog, Cameron, and Visser [8] impose periodic constraints where the robots must meet at specific rendez-vous times in order to share information.

Background

Decentralized - POMDPs

Dec-POMDP [2] is an extension of POMDP for decentralized control domains. A Dec-POMDP is defined with a tuple $< I, S, A, T, R, \Omega, O >$. I is the number of agents, S the set of joint states and $A = \{A_i\}$ the set of joint actions[3]. T :

[3] A state of the problem can be written with a tuple $s = (s_1, ..., s_I)$ such that s_j is the state of robot j. A_i defines the set of actions a_i of robot i.

$S \times A \times S \rightarrow [0;1]$ is a transition function and $R : S \times A \rightarrow \Re$ a reward function. Ω is a set of observations an agent can receive about the environment and $O : S \times A \times S \times \Omega \rightarrow [0;1]$ an observation function. If the global state of the system is collectively totally observable, the Dec-POMDP is reduced to a Dec-MDP.

We can see an MDP [14] as a Dec-MDP where $I = 1$. It is defined with a tuple $< S, A, T, R >$. The goal of MDP planning is to find a sequence of actions maximizing the long-term expected reward. Such a plan is called a policy $\pi : S \rightarrow A$. An optimal policy π^* specifies for each state s the optimal action to execute at the current step assuming the agent will also act optimally at future time steps. The value of π^* is defined by the optimal value function V^* that satisfies the Bellman optimality equation:

$$V^*(s) = \max_{a \in A}(R(s,a) + \gamma \sum_{s' \in S} T(s,a,s')V^*(s')) \quad (1)$$

where γ is the discount factor. Solving a Dec-POMDP is done by computing the optimal joint policy. However the time complexity is NEXP-complete [2], that is incredibly hard. Recent interaction-oriented resolution methods of Dec-POMDPs presented in our related works reduce this complexity.

Distributed Value Functions

Distributed value functions (DVF) have been introduced by [16] as a way to distribute reinforcement learning knowledge through different agents in the case of distributed systems. In a recent paper, we formalize IO resolution of Dec-MDPs with DVFs [10]. In our method, DVF describes the Dec-MDP with two classes: no-interaction class represented as a set of MDPs, one per agent; and the interaction class for close interactions. The Dec-MDP is solved as a collection of MDPs and the interactions between MDPs are considered by passing some information between agents. This leads to a significant reduction of the computational complexity by solving Dec-MDP as a collection of MDPs. Thus, the NEXP complexity of solving a DecMDP is reduced to the complexity of solving a set of MDPs (polynomial), one per agent. Each agent computes locally a strategy that minimizes conflicts, *i.e.* that avoids being in the interaction class. The interaction class is a separate layer solved independently by computing joint policies for these specific joint states.

DVF technique allows each agent to choose a goal which should not be considered by the others. The value of a goal depends on the expected rewards at this goal and on the fact that it is unlikely selected by other agents. In case of permanent communication, an agent i computes its DVF V_i according to :

$$\forall s \in S \quad V_i(s) = \max_{a \in A} \left(R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \right.$$
$$\left. [V_i(s') - \sum_{j \neq i} f_{ij} P_r(s'|s_j) V_j(s')] \right) \quad (2)$$

where $P_r(s'|s_j)$ is the probability for agent j of transitioning from its current state s_j to state s' and f_{ij} is a weighting factor that determines how strongly the value function of agent j reduces the one of agent i. Thus each agent computes strategies with DVF so as to minimize interactions. However when situations of interaction occur, DVF does not handle those situations and the local coordination must be resolved with another technique. For instance joint policies could be computed off-line for the specific joint states of close interactions.

New Distributed Value Functions

DVF provides an interesting way to manage the coordination of decentralized decision makers. However, it assumes a permanent and free communication. In this section, we relax the strong assumptions about communication.

Common Settings

We assume that the local observability of each agent is total so our approach takes place in the Dec-MDP framework. We also assume that each agent shares with the others only its current state. So if the communication never fails, agent i knows at each step t the state $s_j \in S$ of each other agent j. When the communication breaks, the states of other agents are not known at each time step. We make the assumption that $s_j \in S$ is the last known state of another agent j at time t_j. In other words, t_j is the latest time step where the communication between agents i and j succeeded. At the current time t, agent i knows that agent j was at state s_j $\Delta t_j = t - t_j$ time steps ago.

New Distributed Value Functions

Given our limited share of information, the agents cannot exchange information about their value functions. However each robot i can compute all V_j by empathy. Second, to be robust to breaks in communication, we must consider that the states of other agents are not known at each time step. Thus eq. 2 can be rewritten as following:

$$\forall s \in S \quad V_i(s) = \max_{a \in A} \left(R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \right.$$
$$\left. [V_i(s') - \sum_{j \neq i} f_{ij} P_r(s'|s_j, \Delta t_j) V_j(s')] \right) \quad (3)$$

where $P_r(s'|s_j, \Delta t_j)$ is the probability for agent j of transitioning to state s' knowing that j was at state s_j Δt_j time steps ago. In case the communication never fails, the last known state of an other agent j is at current time $t_j = t$ so $\Delta t_j = 0$. We

set:

$$P_r(s'|s_j, \Delta t_j = 0) = P_r(s'|s_j) \tag{4}$$

Thus eq. 3 is a new DVF that can be used with or without permanent communication.

However, one challenge to compute new DVF is the estimation by agent i of the transition probability of another agent j knowing that it was at state s_j Δt_j time steps ago. This could be evaluated with:

$$P_r(s'|s_j, \Delta t_j) = \eta \sum_{\tau \in Traj(s_j, s')} P_r(\tau, \Delta t_j) \tag{5}$$

where η is a normalizing constant, $Traj(s_j, s')$ is the set of trajectories from s_j to s' and $P_r(\tau, \Delta t_j)$ the probability that agent j follows the trajectory τ. The sum over all possible trajectories is the belief that agent j could be at state s' knowing that it was at s_j Δt_j time steps ago. The computation complexity of this probability depends on many factors. First according to the model used, computing the set of trajectories from one state to another in an MDP could quickly become very complex. However, we believe that the complexity could be reduced by using structured models, such as Voronoi diagrams in our robotic exploration case. Second the targeted application could also lead to a simplification of this probability computation, as for instance the exploration case detailed in the next section.

Multi-Robot Exploration

Our approach is primary motivated by a real-world application of mobile robots to explore an unknown environment under breaks in communication constraints. We consider that the state of the robot is known at decision time, that each robot has access to a map updated with all explored areas and to the positions of the others (while the robots can communicate).

New Distributed Value Functions for Multi-Robot Exploration

In case of breaks in communication, the DVF algorithm (eq. 3) is difficult to apply due to the computational complexity of the transition probability $P_r(s'|s_j, \Delta t_j)$. Similarly to [6], we can consider that a robot ignores the information about its neighbors with which the communication failed *i.e.*:

$$\forall s \in S \quad V_i(s) = \max_{a \in A} \Bigg(R(s,a) + \gamma \sum_{s' \in S} T(s,a,s') \\ [V_i(s') - \sum_{j \in \Omega(i)} f_{ij} P_r(s'|s_j) V_j(s')] \Bigg) \tag{6}$$

where $\Omega(i)$ is for the robot i, the set of all the other robots with which i is still in communication. Obviously, in the worst case, this leads to a situation where the robots act as if they were independent.

However, the computation complexity can be reduced in case of an application of exploration. Indeed the estimation of the transition probability (eq. 5) can be rewritten as:

$$P_r(s'|s_j, \Delta t_j) = \eta \sum_{s'' \in S} P_r(s'|s'') P_r(s''|s_j, \Delta t_j) \tag{7}$$

To evaluate successively each probability from one state to another $P_r(s'|s'')$ until the state s_j is reached, a wavefront propagation algorithm can be applied from the last known state s_j of robot j. The obtained values can be consistent with probabilities and represent the future intentions of a robot from its last known state. Anyway this is at reduced efficiency if the communication breaks for a long time.

Computation Details

Algorithm 1: New DVF pseudo-code for agent i

```
/*VI is Value Iteration algorithm                                  */
/*WP is Wavefront Propagation algorithm                            */
begin
    while exploration not finished do
        //One decision step
        Recover newly obtained exploration data
        Recover and communicate its current state s_i
        Update MDP model MDP_i =< S, A, T, R >
        forall j ∈ Ω(i) do
            Receives s_j
        V_emp ← VI (MDP_i, γ)
        forall j ≠ i do
            P_r(*|s_j, Δt_j) ← WP (s_j)
        forall j ≠ i do
            V_j = V_emp
            f_ij = max_s R(s) / max_s V_j(s)
        V_i ← DVF (MDP_i, f_ij, γ, V_j, P_r)
        Follow the policy associated with V_i
end
```

The different steps to compute new DVF in case of exploration are summed up in Algorithm 1 for one agent i. A decision step consists in updating data structures of the MDP model from newly obtained data and computing the policy from DVF.

More details about the MDP model are given in [9]. The robots we consider are homogeneous so the *empathic value function* V_{emp} is computed only once by robot i with the standard value iteration algorithm [1]. To evaluate the transition probability of other robot j, i applies a wavefront propagation algorithm from the current state s_j of robot j. The weighting factor f_{ij} allows to balance the value function with respect to the rewards.

The agent plans continuously, updating its model and policy as it perceives changes in its environment. This allows to update quickly action plan so as to react as soon as possible to the decisions of the others and to information gained *en route*[4]. However, this requires the decision step to be quick enough for on-line use. Given that the model will be updated at each decision step, we use the greedy approach that plans on short-term horizon.

Local coordination

New DVF compute strategies so as to minimize interactions but it does not handle situations when the robots are close to each other that can happen for instance in the starting zone[5] or in some narrow corridor. To solve local coordination, a Multi-agent MDP (MMDP) [5] can be used as the robots concerned by the interaction are close to each other. Joint policies are computed off-line for these specific joint states and followed when local coordination is detected. In our context, these situations can be easily detected by computing the distance between the robot and its partners. When it is inferior to a security threshold (local coordination necessary), this results in different behavior according to the location one to another. In case a robot follows another one closely or if they are face to face, one robot stops and waits for the other to pass. When the robots diverge, none of the robots freeze; when a robot is followed, it moves according to its policy. Additionnaly, an emergency security threshold leads to a retro-traverse movement such that one robot backtracks to let the other pass if they are too close. If the communication fails, the robots act as if they were independant so there is no coordination. But when they are within sensing range, collision avoidance via the anti-collision system is applied as a low-level control instead of local coordination.. Such situations are illustrated in our experiments.

Experimental Platforms

This section describes real and simulated robots.

[4] map is often explored before the robot reached its target.

[5] In our challenge, all the robots must start and return in a specified zone where close interactions will necessarily take place.

Real Robots

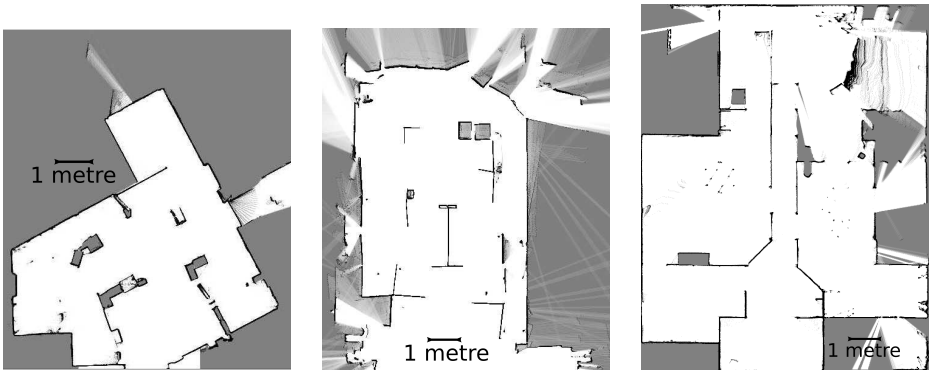
Our Wifibot[6] μ-troopers are 6-wheels mobile robots that embed an Intel Core 2 Duo processor, 2GB RAM and 4GB flash. Each one is equipped with a laser range scanner. The software running on-board is based on a Data Distribution System (DDS) implementation from OpenSplice[7]. This middle-ware allows for several programs to run concurrently, even on different computers. In our architecture, various modules can run asynchronously: Laser acquisition, SLAM, Decision and Mobility. Each robot is independent and has its own modules. The *SLAM module*, based on [20], receives laser readings and provides the other modules with the robot state. The architecture allows the robots to exchange their laser scans and their states. While the communication does not fail, each robot knows the areas explored by the others and updates its local map with local and distant scans. During a break in communication, nothing is exchange between the robots. However, our architecture is robust to communication failures. As soon as a communication between the robots has been re-established, the map is updated by explored areas of the others and their relative states are again exchanged. The *mobility module* implements an advanced point and shoot algorithm, along with a backtrack feature preventing the robot from being stuck. The *decision module* runs asynchronously, computing a new policy every second in average according to Algorithm 1.

Simulated Robots

We use the Stage[8] simulator with an architecture that mimics the real robots. DDS is replaced by an Inter Process Communication shared memory segment. Laser acquisition is simulated by a "laser" virtual sensor. A "position" virtual device simulates both the SLAM module by providing odometric data and the mobility module by executing the point and shoot algorithm. Finally the decision module used on real robots can be used with the simulator without modification.

Experimental Results

In this part we show results from real robot scenarios and simulations where we test various various communication schemes (permanent communication, no communciation and communication breaks) to show the effectiveness of our new DVF when communication is not reliable. These tests are more general than Pentagone or ISR environments [11].



(a) Laboratory experiments. (b) Challenge.

Figure 2: Resulting pixel maps of some areas explored with two robots. Pixels color ranges from black (obstacle) to white (free).

Real Robots

We performed experiments with our two μ-troopers. The videos, available at `http://lmatigno.perso.i` `research`, show the exploration of the robots. Some interesting situations are underlined (global task repartition, local coordination, returning home). Resulting maps of these explored areas are in Fig 2a. Fig. 2b is the resulting map of the challenge. Fig. 3a shows local maps as seen by each robot in which the distant robot is seen by the local one in its local map. In this snapshot of one video, robots split the space to explore different areas: robot1 decides to explore the top of the map and robot2 to explore a corridor. Fig. 3b depicts a local coordination situation successfully resolved: robot1 decides to move while robot2 waits for the other to pass the door. No breaks in communication occurred during these experiments. However, we observed some temporary network errors during other tests and we noticed that our approach resulted in an efficient exploration process. Once the connection had been re-established, the robots coordinated with each other again as our architecture is robust to communication failures.

Simulated Robots

Dense Simulated Environments

To show the benefit of our DVF extension in case of communication breaks, we used two different simulated environments from Stage (Fig. 4). We chose to show results with these complex environments because they are dense. Indeed, in such environments, there are many possible situations of local interaction, and that is the main difficulty to overcome. Local interactions are a good indicator of the quality

[6] www.wifibot.com
[7] http://www.opensplice.com
[8] http://playerstage.sourceforge.net/

of our DVF policy. Using random environments, we would have sparse environments with fewer local interaction situations. The chosen complex environments are most interesting as they show the interest of the DVF. In sparse environments, the DVF would be as much performant but we would not have our local interactions indicator.

Communication Schemes and Strategies

We test 4 cases using various communication schemes and different strategies : no communication *i.e.* the robots are independent and compute strategies using standard value iteration (eq. 1); permanent communication with DVF used by the robots (eq. 2); and communication breaks. In this case, the robots use DVF when communication is available and during the breaks, two methods are compared. First they can be independent during the breaks and each individually explores the environment. This is the method used in [6] and refered as eq. 6. Second the robots can use new DVF (Algorithm 1). During each simulation, 5 breaks take place stochastically, each lasts 25 seconds in office-like1 environment and 40 seconds in office-like2 environment.

Indicator of the Quality of DVF Policy

We plot for each case the time spent to finish the mission (explore the environment and all robots returned to their starting position) and the cumulated mean time spent by all agents in local interactions during one exploration. Indeed, the suboptimality comes from local interaction situations. An optimal joint policy should minimize local interactions, that's why we use local interactions as a good indicator of the quality of our policy. Local interactions are defined as a distance between two robots inferior to 1 meter. During local interactions, local coordination is required if it is possible *i.e.* if the robots communicate.

Results

Results are in Fig. 5. Independent agents (no communication) finished the mission with the higher time since the robots individually explore and each robot covers all the space. Permanent communication gives the fastest exploration and manages to minimize local interactions as robots spread out efficiently. In case of communication breaks, new DVF drastically reduces local coordination compared with the existing approach in the literature (independent agents during breaks) and the exploration time is slightly superior to the permanent communication case. It shows that with new DVF, the agents will manage to coordinate themselves even if the communication breaks. The local interactions indicator shows that our approach avoids many local interaction situations which are particularly hazardous during a break as local coordination is not available but only the anti-collision system.

Moreover, it is interesting to notice that local interactions are more or less important without communication according to the structure of the environment and

especially to the situations of the agents when the communication breaks. Indeed if a break occurs when the agents are distant, they will not come into close interactions. In some environment as in Fig. 4b, local interactions are less frequent even without communication given that the structure of the environment allows independent agents to be seldom close together. However new DVF still reduces local coordination compared with the other approach.

Conclusion

In this paper, we address the problem of multi-robot exploration under communication breaks constraints with an IO resolution of Dec-MDPs. We extend the DVF methodology that assumes permanent and free communication and propose a new DVF to support breaks in communication. We apply this method to multi-robot exploration scenarios, so that each robot computes locally a strategy that minimizes the interactions between the robots and maximizes the space coverage. Experimental results from real-world scenarios and our vice-champion rank at the robotic challenge show that this method is able to effectively coordinate a team of robots during exploration and is robust to communication breaks.

References

[1] Bellman, R. 1957. *Dynamic programming: Markov decision process*.

[2] Bernstein, D. S.; Givan, R.; Immerman, N.; and Zilberstein, S. 2002. The complexity of decentralized control of markov decision processes. *Math. Oper. Res.* 27:819–840.

[3] Beynier, A., and Mouaddib, A.-I. 2005. A polynomial algorithm for decentralized markov decision processes with temporal constraints. In *Proc. of AAMAS*, 963–969.

[4] Beynier, A., and Mouaddib, A.-I. 2006. An iterative algorithm for solving constrained decentralized markov decision processes. In *Proc. of AAAI*, 1089–1094.

[5] Boutilier, C. 1996. Planning, learning and coordination in multiagent decision processes. In *TARK*.

[6] Burgard, W.; Moors, M.; Stachniss, C.; and Schneider, F. 2005. Coordinated multi-robot exploration. *IEEE Transactions on Robotics* 21:376–386.

[7] Canu, A., and Mouaddib, A.-I. 2011. Collective decision- theoretic planning for planet exploration. In *Proc. of ICTAI*.

[8] Hoog, J.; Cameron, S.; and Visser, A. 2010. Autonomous multi-robot exploration in communication-limited environments. In *Towards Autonomous Robotic Systems*.

[9] Matignon, L.; Jeanpierre, L.; and Mouaddib, A.-I. 2012a. Distributed value functions for multi-robot exploration. In *Proc. of ICRA*.

[10] Matignon, L.; Jeanpierre, L.; and Mouaddib, A.-I. 2012b. Distributed value functions for the coordination of decentralized decision makers (short paper). In *Proc. of AAMAS*.

[11] Melo, F. S., and Veloso, M. M. 2011. Decentralized mdps with sparse interactions. *Artif. Intell.* 175(11):1757–1789.

[12] Nair, R.; Varakantham, P.; Tambe, M.; and Yokoo, M. 2005. Networked distributed pomdps: A synthesis of distributed constraint optimization and pomdps. In *Proc. of AAAI*, 133–139.

[13] Powers, M.; Balch, T.; and Lab, B. 2004. Value-based communication preservation for mobile robots. In *Proc. of 7th International Symposium on Distributed Autonomous Robotic Systems*.

[14] Puterman, M. L. 1994. *Markov decision processes*. John Wiley and Sons.

[15] Pynadath, D., and Tambe, M. 2002. Multiagent teamwork: Analyzing the optimality and complexity of key theories and models. *Journal of Artificial Intelligence Research* 16:389–423.

[16] Schneider, J.; Wong, W.-K.; Moore, A.; and Riedmiller, M. 1999. Distributed value functions. In *Proc. of ICML*, 371–378.

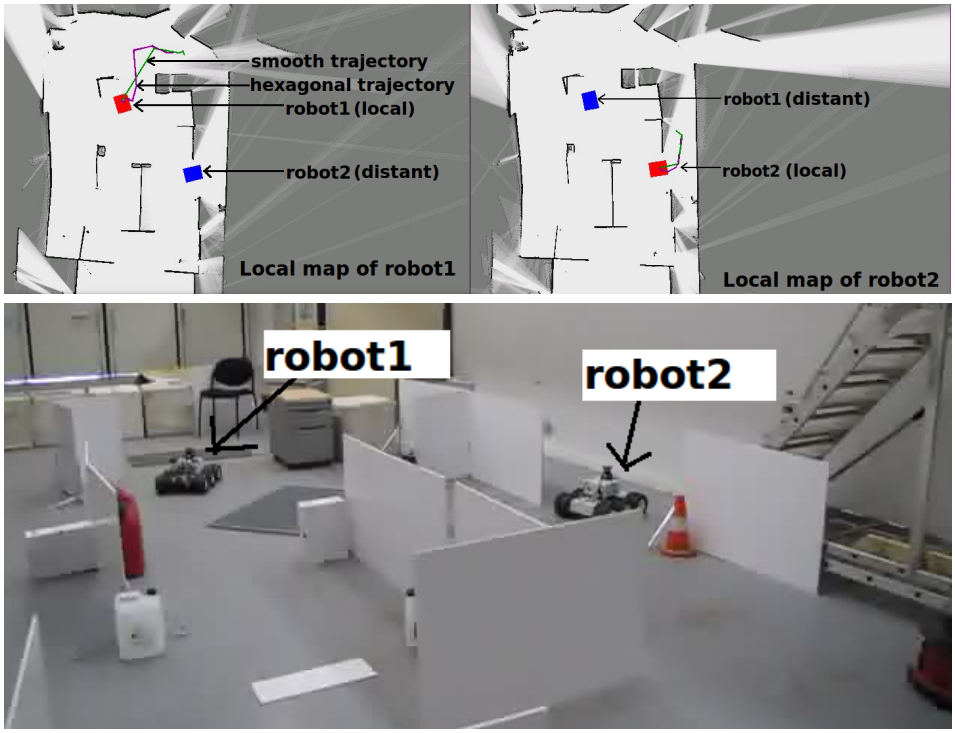
[17] Spaan, M. T. J., and Melo, F. S. 2008. Interaction-driven markov games for decentralized multiagent planning under uncertainty. In *Proc. of AAMAS*, 525–532.

[18] Velagapudi, P.; Varakantham, P.; Sycara, K.; and Scerri, P. 2011. Distributed model shaping for scaling to decentralized pomdps with hundreds of agents. In *Proc. of AAMAS*, 955–962.

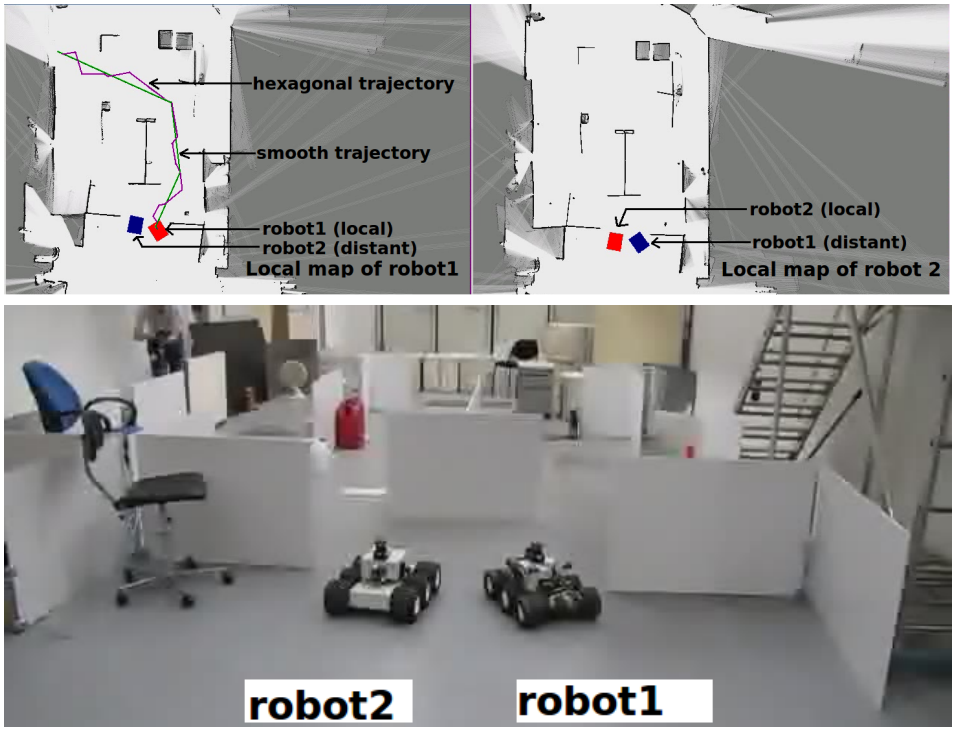
[19] Wurm, K. M.; Stachniss, C.; and Burgard, W. 2008. Coordinated multi-robot exploration using a segmentation of the environment. In *Proc. of IROS*, 1160–1165.

[20] Xie, J.; Nashashibi, F.; Parent, N. M.; and Garcia-Favrot, O. 2010. A real-time robust slam for large-scale outdoor environments. In *17th ITS World Congress*.

[21] Zlot, R.; Stentz, A.; Dias, M.; and Thayer, S. 2002. Multi-robot exploration controlled by a market economy. In *Proc. of ICRA*, volume 3, 3016–3023.



(a) Global coordination.

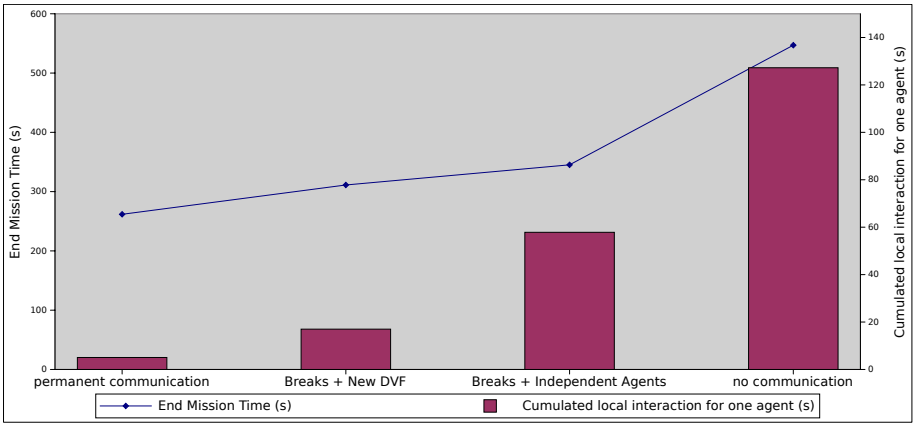


(b) Local coordination.

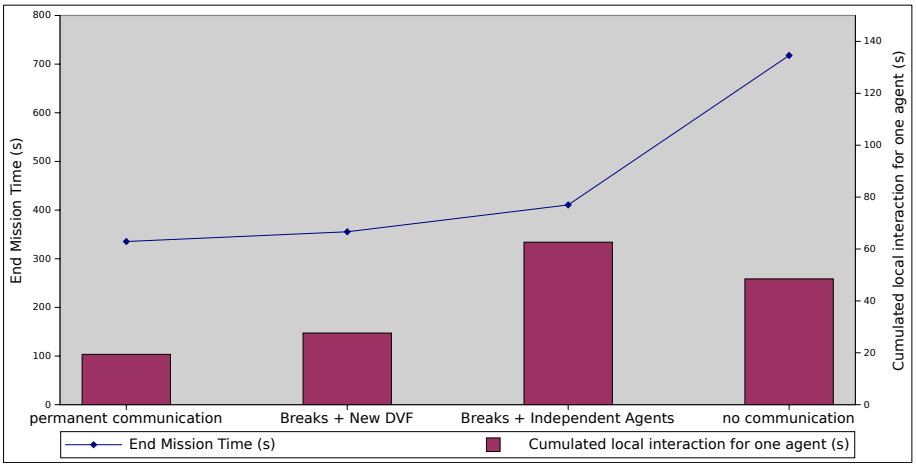
Figure 3: Consistent local maps and video snapshots.

(a) office-like1 (b) office-like2

Figure 4: Simulation environments with starting positions.



(a) office-like1 with 4 robots.



(b) office-like2 with 3 robots.

Figure 5: Results averaged over 5 simulations.