Map Partitioning to Approximate an Exploration Strategy in Mobile Robotics

Guillaume Lozenguez[12], Lounis Adouane[2], Aurélie Beynier[3], Philippe Martinet[2] and Abdel-Illah Mouaddib[1]

Abstract In this paper, we present an approach to automatically allocate a set of exploration tasks between a fleet of mobile robots. Our approach combines a *RoadMap* technique and Markovian Decision Processes (MDPs). We are interested in the problem of exploring an area where several robots need to visit a set of points of interest. This problem induces a long term horizon motion planning with a combinatorial explosion. The *RoadMap* allows us to represent spatial knowledge as a graph of paths. It can be modified during the exploration mission requiring the robots to use on-line computation. By decomposing the *RoadMap* into regions, an MDP allows the leader robot to evaluate the interest of each robot in every single region. Using those values, the leader can assign the exploration tasks to the robots.

1 Introduction

The problem of exploring an environment with a fleet of robots is a persistent topic in mobile robotics [1]. It is difficult to consider the global problematic in a long term horizon but several studies have contributed in different orientations. Indeed, the topic can be decomposed to 2 main parts: planning and controling of the robot's motion [2][3] with localization [4] ; communicating and computing the collaborative exploration strategies [1][5][6]. The notion of *RoadMap* (as a topological map increased by metric informations [7]) appears as an interesting tool to connect the

[1] GREYC, Campus Côte de Nacre, Bd Marchal Juin, BP 5186, 14032 Caen Cedex France
e-mail: [firstname.lastname]@info.unicaen.fr

[2] LASMEA, Campus des Cézeaux, 24 Avenue des Landais, 63177 Aubiere Cedex France
e-mail: [firstname.lastname]@lasmea.univ-bpclermont.fr

[3] LIP6, University Pierre & Marie Curie, Boîte courrier 169, 4 place Jussieu, 75005 Paris France
e-mail: [firstname.lastname]@lip6.fr

Supported by the Nationnal Reserch Agency of France (ANR) through the *R-Discover* project

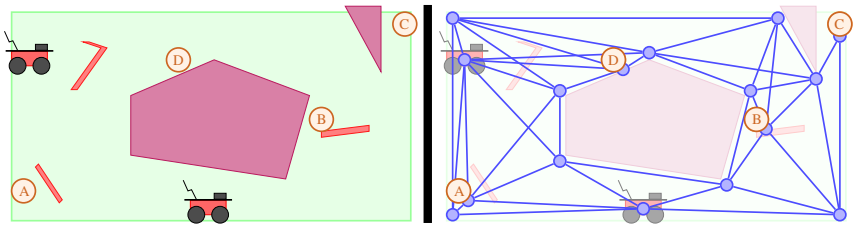

Fig. 1 Exploration problem with 4 points of interest $\{A, B, C, D\}$ and the attached *RoadMap*.

sensors control and decision making [8]. The approach in this paper proposes the use of the *RoadMap* in on-line decision making considering a long-term horizon.

Our research is employed in a project named R-Discover that aims at exploring an external area using a fleet of robots. An Unmanned Aerial Vehicle (UAV) takes several pictures of an area and then a fleet of ground robots are set to refine knowledge about this area. Indeed, the pictures allow us to build a first map regarding detected obstacles. Next, an operator defines key positions which must be visited to increase the map definition as a set of points of interest in the map (Fig. 1).

The main concern of the paper is based on the robots' capacities to cooperate in order to calculate and adapt exploration strategies along the mission. Robots are equipped with communication devices efficient in a given radius. In this paper, we focus on the particular steps of the mission where two or more robots of the fleet can communicate. We suppose that robots are able to share their knowledge. The present robot with the higher level in a defined hierarchy is set to be the current leader. We are interested in the capacity of the current leader to re-allocate the local set of points of interest $I = \{A, B, C, \ldots\}$ between the present robots in few seconds.

Allocating the set of points of interest is computed in a way that maximizes the sum of individual expected gains. But the complexity of evaluating the interest of a robot in visiting a sub-set of points of interest does not permit a computation of an optimal solution for real size problems $|I| > 20$ (Section 2.2). To bypass the complexity of finding a solution on-line, heuristics are used to partition the *RoadMap* which allows the robots to plan and reason over regions instead of the set of points of interest. The approach permits to solve the problem of exploring an area with a fleet of robots with planning under uncertainty over long term horizon using methods to minimize the on-line computation time.

2 Working Context

The used architecture presented in the next section allows us to separate the robot locomotion problem from the deliberative aspect. This way, we concentrate on the problem of computing the exploration policies using several levels of abstraction in order to allocate the points of interest between the present's robots.

2.1 The Robot's Control Architecture

Each robot is composed of two modules. The first is reactive and permits the robot to move between two positions. The second is deliberative and aims to organize the tasks and gives the decision to the reactive module (Fig. 2).

The reactive module controls movements according to events of low importance such as little discovered obstacles to avoid. A hybrid multi-controller [3] is used for the navigation of the mobile robot in cluttered environments. This architecture is based on a flexible switching between different atomic controllers as attraction to a target or obstacle avoidance. That allows the robot to reach a position while avoiding obstacles.

In this paper, we are not interested in developing the reactive module. In fact, we are interested in describing the deliberative module. A task (Fig. 2) matches a target position to reach by avoiding obstacles. A Probabilistic *RoadMap* [2] is defined as a graph $< W, P >$ where W is the set of way-points (nodes) and P is the set of paths (edges). The way-points set W is composed of the points of interest I in addition to the environment way-points (points around the known obstacles)(Fig. 1). The *RoadMap* is assumed fully connected.

$$RoadMap = \{W, P\}, \quad where: \quad W = \{w_1, ..., w_k\}$$
$$P = \{(w_1,\ w_2,\ \overrightarrow{v},\ c,\ u) \mid w_1, w_2 \in W, \quad \overrightarrow{v} \in \mathbb{R}^2, \quad c \in \mathbb{R}, \quad u \in [0,1]\}$$

Each path p is defined by the current w_1 and the targeted w_2 way-points. A vector $\overrightarrow{v}$ gives the relative position of w_2 from w_1. The attribute c is the associated cost; it depends on the distance and the quality of the path. The attribute u is the probability to reach a target position without crossing important obstacles.

Structuring knowledge of collision-free connectivity in a *RoadMap* permits to use graph algorithms like A^* to plan the movements of an agent or a fleet of agents [6]. Graph theory does not directly fit to the stochastic aspect to find an optimal policy in uncertain environments. However it is possible to combine the *RoadMap* and Markovian Decision Processes. It was used to improve path finding by minimizing the movement cost and considering collision safe paths [8]. Due to non-deterministic actions, we propose to compute robot's policies using Markov Decision Processes (*MDP*) [9], where the MDP model is built from the *Roadmap* elements.

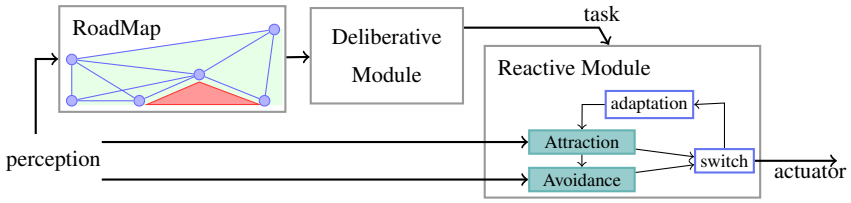

Fig. 2 The Robot Architecture Schema.

2.2 The Markovian Decision Processes

An MDP is defined as a tuple $< S, A, t, r >$ with S and A respectively, the state and the action sets that define the system and its control possibilities. t is the transition function defined as $t : S \times A \times S \rightarrow [0,1]$ that gives the probability $t(s,a,s')$ to reach the state s' from s by doing action $a \in A$. The reward function r is defined as $r : S \times A \rightarrow \mathbb{R}$, $r(s,a)$ gives the reward obtained by executing a from s.

A policy function $\pi : S \rightarrow A$ assigns an action to each system state. Optimally solving an MDP consists in searching an optimal policy π^* that maximizes the expected gain. π^* maximizes the value function of Bellman equation [10] defined on each state. For a policy π:

$$V^{\pi}(s) = r(s,a) + \gamma \sum_{s' \in S} t(s,a,s') V^{\pi}(s'), \quad a = \pi(s)$$
$$V^{\pi^*}(s) = \underset{a \in A}{argmax}\left(r(s,a) + \gamma \sum_{s' \in S} t(s,a,s') V^{\pi^*}(s') \right)$$

The γ parameter in $[0,1]$ balances the importance between future and immediate rewards. In case of finite horizon problems, as visiting a set of points, γ is set to 1.

Several studies [4][5] use MDPs in mobile robotics. In routing problem, an action is added for each path p ; transitions and rewards are done considering the values u and c linked to p (Section 2.1). To visit a set of points, an MDP state needs to include the set of already visited points of interest I' and the number of states increases exponentially regarding the total number of points of interest ($|S| > 2^{|I|}$). We can not consider a problem with more than 20 points of interest to solve it on-line.

2.3 MDPs Decomposition

Decomposition permits us to decrease the complexity of the policy computation by building a hierarchy between local problems and a global solution. It is particularly efficient in spatial problems as it is based on the topological aspect of transitions. The idea of Decomposed MDPs is to aggregate strongly connected states together in sub-MDPs to compute the policy in a distributed way [11][12]. Several policies are computed for each sub-MDP depending on neighboring parameters values.

The problem of computing an optimal graph partition is known to be NP-Complete [13][14]. In case of Decomposed MDPs, the optimal partition on S is the partition that allows to compute the policy the most quickly possible. Independent sub-MDPs are desired in order to decrease the number of policies computation in each sub-MDP. Generally, the criteria is to built partitions as balanced as possible by minimizing connexions between sub-MDPs [15][16].

The presented approach is based on a greedy decomposition (Section 3.2) of the *RoadMap* in order to build a global abstract MDP on the regions set. This global MDP allows robots to evaluate their interest of exploring each of the regions in order to allocate the mission between them (Section 3.3).

3 The Deliberative Module

During the mission, robots meet each other. At this moment, present robots are able to communicate in order to merge their knowledge. We are interested in how the current leader can build, on-line, a new partition of the updated *RoadMap* and allocate the exploration tasks to the present robots. All points of interest need to be allocated while minimizing the sum of the expected movement costs of all the robots. The solution must be able to handle 3 communicating robots, up to 120 targets to visit and an exploration area including around a thousand way-points.

3.1 The RoadMap Partition

After updating its *RoadMap* regarding transmitted information from present robots, the current leader partitions the *RoadMap* into regions in order to re-allocate the set of regions of interest between the present robots.

We search the k-partition that maximizes the ratio between the number of paths contained in each region over the number of paths which connect two regions. We denote a partition as $\Phi_k = \{R_1 \dots R_k\}$, defined on k regions of the *RoadMap*. A partition covers the way-points set with no intersection between two regions:

$$\bigcup_{R_i=\Phi_k} R_i \quad = W, \qquad \forall R_i, R_j \in \Phi_k, \qquad (\ R_i \neq R_j\ ) \Rightarrow (\ R_i \cap R_j = \emptyset\ )$$

We define $Input(\Phi_k) \subset P$ as the set of all internal paths contained in regions. In contrast, the set $Output(\Phi_k) \subset P$ contains all intersected paths by the current partition.

$$Input(\Phi_k) = \{p(w_1, w_2, \overrightarrow{v}, c, u) \mid \exists R_i \in \Phi_k, \quad (w_1, w_2) \in R_i \times R_i\}$$
$$Output(\Phi_k) = \{p(w_1, w_2, \overrightarrow{v}, c, u) \mid \exists R_i, R_j \in \Phi_k, \quad R_i \neq R_j, \quad (w_1, w_2) \in R_i \times R_j\}$$

Finally we search the optimal partition Φ_k^* defined on the *RoadMap* which maximizes the criteria $\frac{|Input(\Phi_k)|}{|Output(\Phi_k)|}$. Graph partitioning is known to be NP-complete and we need to compute a solution on-line during the mission. Therefore, we use a greedy heuristic. We choose to build regions incrementally by adding the way-points which maximize the criteria for a current region.

3.2 The Greedy Heuristic

Similarly to $Input(\Phi_k)$ and $Output(\Phi_k)$, during the construction of the partition Φ_k we define $Input(R_i, w)$ and $Output(\Phi_k, R_i, w)$ the sets of paths that connect a region R_i to a way-point w and that connect w to way-points not contained yet in the current Φ_k. Starting with a given way-point w_0, Algorithm 1 builds the region R_i by select-

ing the way-points which maximize $criteria(\Phi_k, R_i, w) = \frac{|Input(R_i,w)|}{|Output(\Phi_k,R_i,w)|}$. In case of no way-point w has a criteria value $criteria(\Phi_k, R_i, w)$ up to a bound b (fixed to 1 in our study), a new region begins with the closest free way-point to w_0.

Algorithm 1 Greedy Partitioning

Require: $RoadMap\ M = \{W, P\}, \quad w_0 \in W, \quad b \in \mathbb{R}^+, \quad \Phi_k \leftarrow \emptyset$
  while $\bigcup_{R_i \in \Phi_k} R_i \neq W$ **do**
    $R' \leftarrow \emptyset$
    choose w' the closest way-point to w_0 in term of distance where $w' \in W - \bigcup_{R_i \in \Phi_k} R_i$
    repeat
      $R'.add(w')$
      choose w' that maximizes $M.criteria(\Phi_k, R', w')$ and $w' \in W - \bigcup_{R_i \in \Phi_k} R_i$
    until $M.criteria(\Phi_k, R', w') > b$
    $\Phi_k.add(R'), \quad (k \leftarrow k+1)$
  end while
  return Φ_k

Furthermore, we control the expected size of regions by balancing $Output(\Phi_k, R_i, w)$ with a parameter $e(R_i)$ that simulates future paths inside R_i. The effect of $e(R_i)$ is bounded by the number of $Output(\Phi_k, R_i, w)$. We define the criteria as:

$$criteria(\Phi_k, R_i, w) = \frac{|Input(R_i, w)|}{|Output(\Phi_k, R_i, w)| - \min(\ |Output(\Phi_k, R_i, w)|,\ e(R_i)\ ) + \varepsilon}$$

This way, negative values of $e(R_i)$ force the selection to close the region by increasing the number of cut paths. At each step, $e(R_i)$ is defined inversely proportional to the number of points of interest added to the region. We denote by d the desired number of points of interest contained in a region. Furthermore, k^* denotes the expected number of regions ($k^* = |I|/d$). In fact, the greedy partitioning does not guaranty that $k = k^*$.

$$e(R_i) = \frac{d - |R_i \cap I|}{d} \cdot \frac{|P|}{|W|}, \quad d \in \mathbb{N}^+$$

3.3 The Global MDP and Region Allocation

We will explain how MDPs are used to model a global exploration problem. The global MDP allows the leader to valuate each robot interest in exploring a sub-set of regions. From a partition $\Phi_k = \{R_1 \dots R_k\}$, a *RoadMap* and a set I of points of interest to allocate, we define a set of regions of interest $J \subset \Phi_k$ as the set of all regions with at least one point of interest. A state $s = (R_s,\ J_s)$ of the global MDP includes the region R_s where the robot is positioned and $J_s \subset J$ the set of explored regions. A specific state called *block* is added to represent the situation where an unknown

obstacle prevents the robot from reaching a task. Falling in the *block* state means that the robot needs to recalculate the global computed policy. This situation comes with important updates of the *RoadMap* that modifies the region configuration. In this model, when a robot is in a state (R_s, J_s), its possible actions are moving to an adjacent region R'_s or exploring the current region R_s.

$$S = \{(R_s, J_s) \mid R_s \in \Phi_k, \quad J_s \subseteq J\} \cup \{block\}$$
$$A = \{goto_{R'_s} \mid R'_s \in \Phi_k\} \cup \{explo_{R_s} \mid R_s \in J\}$$

In the first case the robot ends up in state (R'_s, J_s), and in the second case it ends up in state $(R_s, J_s \cup \{R_s\})$. We denote $suc(s,a)$ the reached state if the execution of the action a is successful. The transition function $t(s,a,s')$ gives the probability to reach the state $suc(s,a)$ or to get *blocked*.

Optimal transition and reward evaluation depend on the shape of each region. It depends on the local region policies and the crossed way-points. The computation time does not permit us to compute all local policies so we chose to approximate the transition and the reward functions. The approximation is done proportionally to the values of uncertainty u and the cost c linked to each path p of the *RoadMap* and the number of included points of interest $|R_i \cap J|$ of the explored region R_i.

Using the *ValueIteration* algorithm [9] on the global MDP allows the robots to compute an abstract policy as moving between regions and exploring them. We search to automatically allocate a set of regions to explore to each robot of the n present robots at a communication step of the mission.

We want to find the best allocation $\mathbb{J}_n^* = \{J_0, \ldots, J_n\}$ where each $J_i \in \mathbb{J}_n$ matches the set of regions allocated to the robot Ag_i. The optimal allocation maximizes the sum of the robots expected gains in the $n^{|J|}$ possible allocation. Knowing the set R_{Ag_i} of the actual regions of the robot Ag_i, the computed policy π^* of the global MDP and the value function of bellman $V^{\pi}(s)$, we search:

$$\underset{\mathbb{J}_n^* \in \{\mathbb{J}_n^0, \ldots, \mathbb{J}_n^{|J|}\}}{argmax} \quad \left(\quad \sum_{i=0}^{n} V^{\pi^*}(R_{Ag_i}, J_i) \quad \right)$$

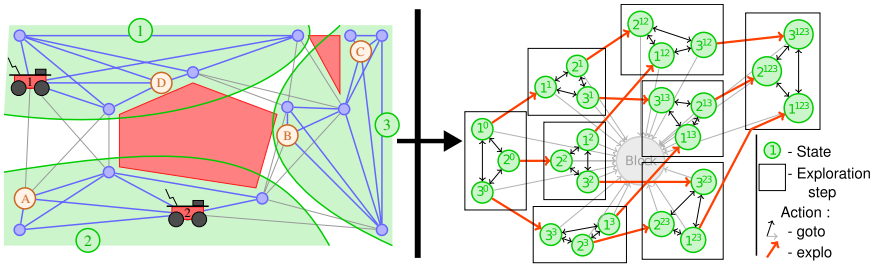

Fig. 3 A partitioned *RoadMap* and the attached global MDP. For example, a state 2^{13} means that the robot is in the region 2 and the regions 1 and 3 are explored

By considering up to 3 robots and 12 regions ($n \leq 3$ and $k* \leq 12$) it is possible to test all the set of n-allocations. The leader, for each possible allocation $\mathbb{J}_n^j$, computes the sum of expected gain for each robot Ag_i ($V^{\pi^*}(R_{Ag_i}, J_i^j)$) and holds $\mathbb{J}_n^*$ with the maximum sum. The architecture of a *RoadMap* with multi-level MDPs permits to consider more than 3 robots for on-line computations. However, the calculation of $\mathbb{J}^*$ by an exhaustive way limits us to consider only few robots in this paper.

4 Experiments

A visual representation of the built partitions (Fig. 4) allows us to conclude: when the environment is more structured with a coherent expected region size, the algorithm builds a partition closer to the expected one. Otherwise, we observe intersected regions in free space environments. It is due to the non-consideration of the path cost in the used criteria. This phenomenon is reduced in cluttered environment.

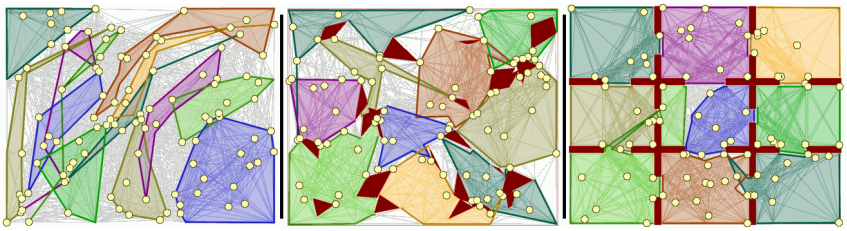

Fig. 4 Examples of partitions built from differently structured environments (RoadMaps)

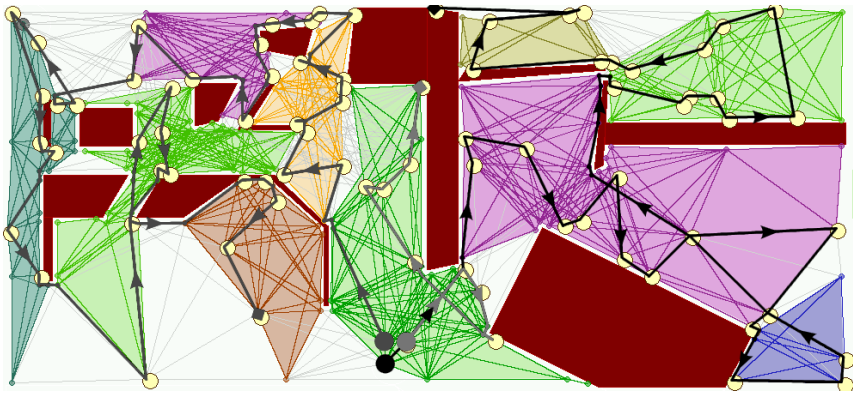

Fig. 5 The 3 likely trajectories following the computed policy ($n = 3$, $|I| = 80$ and $d = 10$). Each robot computes its local policies for each crossed region in a similar way to Section 3.3. Local policies are oriented by the current interest of neighbors regions evaluated in the Global MDP.

$\|I\|$	k	R_i sizes bounds	time (s)	allocation
5	1.13	$4.382-4.968$	0.058822	$1-0-0$
20	3.414	$3.428-8.126$	0.047248	$2-1-0$
40	5.6	$4.12-10.144$	0.0526601	$3-2-1$
60	7.708	$4.292-11.44$	0.11589	$4-2-1$
80	9.426	$4.728-12.742$	0.339208	$5-3-1$
100	10.634	$4.356-14.192$	0.897122	$6-3-1$
120	12.422	$3.744-14.81$	4.9466	$6-4-2$

Table 1 This table shows, for a selected set of experiments (function of the number of points of interest $|I|$), the average number of built regions (k), the average bounds of regions sizes (in regard of the number of contain points of interest), the average computing time and the average allocated number of regions.

(a)

number of region k	k^*-3	k^*-2	k^*-1	k^*	k^*+1	k^*+2	k^*+3	k^*+4	k^*+5	k^*+6
experiment (%)	0.0	0.13	3.76	21.87	48.44	23.15	2.56	0.09	0.01	0.0

(b)

worst region size	8^-	9	10	11	12	13	14	15	16	17	18	19	20
experiment (%)	7.78	11.0	12.1	11.6	12.5	13.3	13.6	9.21	5.13	2.42	0.88	0.26	0.12

Table 2 A presentation of experiments regarding the numbers of built regions k in function of the expected number k^* (a) and the number of contained points of interest in the worst region (b). The percentages are calculated from experiments with between 5 and 120 random points of interest.

We are interested in evaluating this approach in regard to the number of regions, their sizes and the time needed to compute the global policy and allocate the set of regions of interest. The considered problem involved up to 120 points of interest, 3 present robots and a desired number of 10 points of interest by region ($|I| \leq 120$, $n = 3$ and $d = 10$). Our experimentation generates targets randomly (Fig. 5) (500 random generations for different numbers of points of interest $|I|$ growing by 5). We use an Intel Core2 Quad CPU Q9650 at 3.00GHz to compute the partition, the global policies and the region allocation.

Table 1 and Table 2 present some experimentation results that validate the control of the expected number of regions, the numbers of points of interest in a region and the associated computation time. MDP sizes and computation time grow exponentially with the number of regions k or, with the number of points of interest in a region (in local MDPs). We notice that the allocations are unbalanced, it is normal in regard to the placement and shapes of obstacles. Indeed, the intention was to minimize the sum of robots movement cost.

Experiments with few points of interest in regard to the number and the structure of obstacles lead to a partition with more regions than expected ($k > k^*$) and worst regions which have a lower size than desired size d (Table 2). That denotes the capacity of using the shapes of existing obstacles to built different regions.

5 Conclusion and Future Work

This paper presents a decision making architecture for mobile robots sharing an exploration mission. It is difficult for a leader to evaluate the allocations of many tasks between robots. The study is based on knowledge organized in a *RoadMap* and a solver based on a abstract MDP. Indeed the problem size and the constraint of

on-line computation impose to decompose the input data. A greedy decomposition is instantaneous regarding the computation of decision policies. We have demonstrated how a greedy decomposition permits to perform on-line a multi-agent long horizon problem in a stochastic domain by controlling the number of tasks in regions and the number of regions.

In future work, the partitioning quality will be improved by using algorithms based on finding minimal cuts. The idea is to converge to a local optimum from the first greedy decomposition and assume other characteristics as disjunction between regions. Furthermore, adding cooperation in the local MDPs solving will permit several robots to explore a region together. But, that will impact the Global MDP where few robots can explore parts of a same region.

References

1. W. Burgard, M. Moors, C. Stachniss, and F. Schneider, "Coordinated multi-robot exploration," *IEEE Transactions on Robotics*, vol. 21, pp. 376–386, 2005.
2. L. Kavraki, P. Svestka, J. claude Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," in *IEEE International Conference on Robotics and Automation*, 1996, pp. 566–580.
3. L. Adouane, "Hybrid and safe control architecture for mobile robot navigation," in *9th Conference on Autonomous Robot Systems and Competitions*, Portugal, May 2009.
4. A. F. Foka and P. E. Trahanias, "Real-time hierarchical pomdps for autonomous robot navigation," *Robotics and Autonomous Systems*, vol. 55, no. 7, pp. 561–571, 2007.
5. F. Teichteil-Königsbuch and P. Fabiani, "Autonomous search and rescue rotorcraft mission stochastic planning with generic dbns," in *IFIP AI*, 2006, pp. 483–492.
6. O. Bayazit, J. Lien, and N. Amato, "Swarming behavior using probabilistic roadmap techniques," in *Swarm robotics: SAB international workshop*, Santa Monica, USA, July 2004.
7. B. Kuipers and Y. tai Byun, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations," *Journal of Robotics and Autonomous Systems*, vol. 8, pp. 47–63, 1991.
8. R. Alterovitz, T. Siméon, and K. Goldberg, "The stochastic motion roadmap: A sampling framework for planning with markov motion uncertainty," in *Robotics: Science and Systems*, 2007.
9. M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1994.
10. R. Bellman, "A markovian decision process," *Journal of Mathematics and Mechanics*, vol. 6, pp. 679–684, 1957.
11. T. Dean, S. hong Lin, and S. hong Lin, "Decomposition techniques for planning in stochastic domains," in *14th International Joint Conference on Artificial Intelligence*, 1995.
12. C. Boutilier, T. Dean, and S. Hanks, "Decision-theoretic planning: Structural assumptions and computational leverage," *Journal of Artificial Intelligence Research*, vol. 11, pp. 1–94, 1999.
13. M. R. Garey, D. S. Johnson, and L. Stockmeyer, "Some simplified np-complete problems," in *6h Symposium on Theory of Computing*. New York, NY, USA: ACM, 1974, pp. 47–63.
14. C.-E. Bichot and P. Siarry, *Graph Partitioning*. Wiley-ISTE, 2011.
15. R. Parr, "Flexible decomposition algorithms for weakly coupled markov decision problems," in *14th Conference on Uncertainty in Artificial Intelligence*, 1998, pp. 422–430.
16. R. Sabbadin, "Graph partitioning techniques for markov decision processes decomposition," in *15th Eureopean Conference on Artificial Intelligence*, 2002, pp. 670–674.