Applying Bayesian Learning to Multi-Robot Patrol

David Portugal, Micael S. Couceiro and Rui P. Rocha

Abstract— **Performing a patrolling mission with multiple mobile robots is a challenging task that requires effective coordination between agents. While predefined patrol circuits may lead to suitable routing performance, their deterministic nature eases the task of potential intruders. Therefore, the need to propose probabilistic strategies becomes evident.**

In this paper, a new multi-robot patrolling strategy is proposed, in which concurrent learning agents adapt their moves to the state of the system at the time, using Bayesian decision. When patrolling a given site, each agent evaluates the context and adopts a reward-based learning technique that influences future moves. Experiments show the potential of the approach, which outperforms several other state-of-the-art strategies.

I. INTRODUCTION

Security applications are a fundamental task with unquestionable impact on society. Combining this fact with the technological evolution observed in the last decades, it becomes clear that robot assistance can be a valuable resource in monotonous, repetitive and dangerous missions by taking advantage of robots' expendability.

One of such missions is multi-robot patrolling, which requires agents to coordinate their decision-making to visit every position in the environment (or at least those that need surveillance) so as to achieve collective optimal performance. Despite its high potential utility in security applications, only recently the Multi-Robot Patrolling Problem (MRPP) has been rigorously addressed using principles of task allocation [1], graph theory [2], market-based coordination [3], game theory [4], Markov decision processes [5], artificial forces [6] and others.

Several contributions to the MRPP at a theoretical level have also been presented and it has been shown that the problem is NP-Hard [2], [7]. Within all the strategies pursued so far, the creation of adaptive behaviors that allows agents to learn how to effectively patrol a given scenario are the more promising. Moreover, such adaptability fosters the unpredictability principle in a way that intruders are unable to anticipate patrolling trajectories. Nevertheless, the use of such techniques are far from being straightforward. Certain works in this field have adopted machine learning methods aiming to adapt agents' behavior. For instance, the work of Ishiwaka *et al.* [8] proposed reinforcement learning to predict the location of teammates as well as the movement direction to a common target. Another pioneering approach was proposed by Santana *et al.* modeling the MRPP as a Q-learning problem in an attempt to allow automatic adaptation of the agents' strategies to the environment [9]. In brief, agents have a probability of choosing an action from a finite set of actions, having the goal of maximizing a long-term performance criterion, in this case, node idleness. Two reinforcement learning techniques, using different communication schemes were implemented and compared to non-adaptive architectures. Although not always scoring the best results, the adaptive solutions are superior to other solutions compared in most of the experiments. The main attractive characteristics in this work is distribution (no centralized communication is assumed) and the adaptive behavior of agents, which is usually highly desirable in this domain.

Alternatively to reinforcement learning, some strategies have been using stochastic approaches that benefit from probabilistic decision-making to overcome the deterministic nature of classic patrolling applications. For instance, in [5] the patrolling problem is casted as a multi-agent Markov decision process, where reactive and planning-based techniques are compared. The authors concluded that both perform similarly, with the latter being slightly superior in general, since it looks further ahead than the former, which is purely local. However, the reactive technique runs much faster, suggesting that a simple and computationally cheaper approach can be used in many applications, instead of more complex strategies which only perform slightly better. Chen and Yum [10] also formulated the problem as a Markov decision process and proposed a patrol routing strategy under a finite horizon approximation.

In this work, a new distributed and adaptive approach for multi-robot patrol is proposed. Each robot decides its local patrolling moves online, without requiring any central planner. Decision-making is based upon Bayesian reasoning on the state of the system, considering the history of visits and teammates actions, so as to promote effective coordination between patrolling agents. Experimental results illustrate the advantages of using the proposed technique, when compared to several state-of-the-art strategies.

II. IDLENESS CONCEPT

In this work, the problem of efficiently patrolling a given environment with an arbitrary number of robots is studied. Agents are assumed to have an *a priori* representation of the environment and, in order to easily assess the topology of its surroundings, a graph extraction algorithm is adopted

This work was supported by PhD scholarships (SFRH/BD/64426/2009) and (SFRH/BD/73382/2010), the CHOPIN research project (PTDC/EEA-CRO/119000/2010) and by the ISR-Institute of Systems and Robotics (project PEst-C/EEI/UI0048/2011), all of them funded by the Portuguese science agency "Fundação para a Ciência e a Tecnologia" (FCT).

All authors are with the Institute of Systems and Robotics (ISR), University of Coimbra (UC), Pólo II, 3030-290 Coimbra, Portugal, e-mail: {davidbsp,micaelcouceiro,rprocha} at isr.uc.pt

978-1-4799-0880-6/13/$31.00 ©2013 IEEE

to obtain an undirected navigation graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. $\mathcal{G}$ is composed of vertices $v_i \in \mathcal{V}$ and edges $e_{i,j} \in \mathcal{E}$, where each vertex represents a specific location that must be visited regularly and each edge represents the connectivity between these locations. The MRPP is therefore reduced to coordinate robots in order to frequently visit all $v_i \in \mathcal{G}$, ensuring the absence of atypical situations with regard to an optimization criterion. This criterion should be defined to enable comparison of performance of different patrolling algorithms. Diverse metrics have been previously proposed to access the effectiveness of multi-robot patrolling strategies. Typically, these are based on the idleness of the vertices, the frequency of visits or the distance traveled by agents [11]. In this work, the first one has been considered [12], given that it measures the elapsed time since the last visit from any agent in the team to a specific location. The idleness metric uses time units, which is particularly intuitive to analyze.

The instantaneous idleness of a vertex $v_i \in \mathcal{V}$ in time step t is defined as:

$$\mathcal{I}_{v_i}(t) = t - t_l, \tag{1}$$

where t_l corresponds to the last time instant when the vertex v_i was visited by any robot of the team. Consequently, the average idleness of a vertex $v_i \in \mathcal{V}$ in time step t is defined as:

$$\overline{\mathcal{I}_{v_i}}(t) = \frac{\overline{\mathcal{I}_{v_i}}(t_l) \cdot C_i + \mathcal{I}_{v_i}(t)}{C_i + 1}, \tag{2}$$

where C_i represents the number of visits to v_i. Finally, in order to obtain a generalized performance measure, the average idleness of the graph $\mathcal{G}$ ($\overline{\mathcal{I}_{\mathcal{G}}}$) is defined as:

$$\overline{\mathcal{I}_{\mathcal{G}}} = \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} \overline{\mathcal{I}_{v_i}}, \tag{3}$$

where $|\mathcal{V}|$ represents the cardinality of the set $\mathcal{V}$. In the beginning of the experiments, it is assumed that for all $v_i \in \mathcal{V}$, $\mathcal{I}_{v_i}(0) = 0$, as if every vertex had just been visited at the beginning of the mission. Hence, there is a transitory phase in which the $\overline{\mathcal{I}_{\mathcal{G}}}$ values tend to be low, not corresponding to the reality in a steady-state phase. For this reason, the final $\overline{\mathcal{I}_{\mathcal{G}}}$ value is measured only after convergence to the steady-state.

The patrolling problem with R robots can be described as the problem of finding a set of R paths that visit all vertices $v_i \in \mathcal{V}$ of $\mathcal{G}$, with the overall team goal of minimizing $\overline{\mathcal{I}_{\mathcal{G}}}$. Note, however, that such paths are computed online and locally during the mission, in order to adapt to the system's needs.

III. Concurrent Bayesian Learning Strategy

In a previous work of the authors [13], simple Bayesian-based techniques to tackle the MRPP were studied. Even though the results obtained were satisfactory, two main drawbacks were identified: a uniform prior distribution was adopted, assuming that all decisions were equiprobable; and the likelihood distributions were immutable, representing a fixed function of random variables.

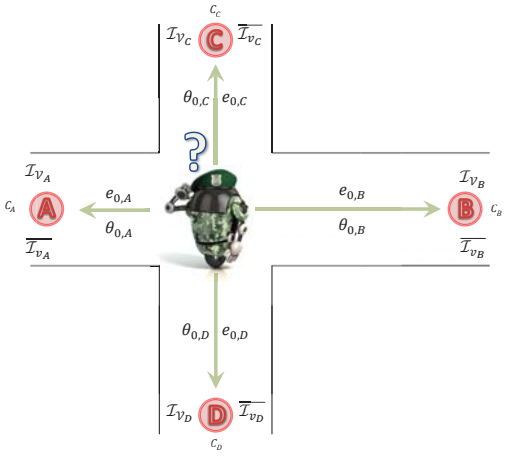


Fig. 1: Illustration of a patrolling decision instance.

In this work, the previous Bayesian models proposed are extended with likelihood reward-based learning and continued prior update. More specifically, the model represents the decision of moving from one vertex of the graph to another. For β neighbors of the current vertex v_0, where $\beta = deg(v_0)$[1], the model is applied β times. Each decision is considered independent and the agents have the ability to choose the action which has the greatest expectation of utility, weighted by the effects of all possible actions. Consequently, each robot's patrol route is built progressively, at each decision step, adapting to the system's needs, *i.e.*, aiming at minimizing $\overline{\mathcal{I}_{\mathcal{G}}}$. In the next section, more details on the Concurrent Bayesian Learning Strategy (CBLS) to solve the MRPP are presented.

A. Distribution Modeling

As stated before, when reaching a vertex v_0 of the navigation graph $\mathcal{G}$, each robot is faced with a decision stage, where it must decide the direction it should travel next (*cf.* Fig. 1). To that end, two fundamental random variable are defined. The first one is boolean and simply represents the act of moving (or not) to a neighbor vertex v_i:

$$move_i = \{true, false\}, \tag{4}$$

while the second one is called *arc strength* $\theta_{0,i}$, which represents the suitability of traveling to a neighbor v_i using the arc that connects v_0 to v_i:

$$\theta_{0,i} \in \boldsymbol{\theta}; \quad 0, i \in \mathbb{N}_0; \quad \text{and} \quad |\boldsymbol{\theta}| = 2|\mathcal{E}|. \tag{5}$$

Note that $\mathcal{G}$ is an undirected graph, where an edge $e_{j,k}$ represents a connection from v_j to v_k and vice versa. $e_{j,k}$ has an edge cost or weight $|e_{j,k}| = |e_{k,j}|$, given by the distance between the two vertices. Nevertheless, the term "arc" instead of "edge" is used intentionally, since it implies

[1]The degree (or valency) of a vertex of a graph is the number of edges incident to the vertex.

a direction of traveling. In a situation where an agent is at v_j, it will look for the suitability of traveling to v_k, given by $\theta_{j,k}$. Under those circumstances, the suitability of traveling in the opposite direction is not relevant, thus $\theta_{j,k} \neq \theta_{k,j}$. As a consequence, the set $\boldsymbol{\theta}$ has a population of $2|\mathcal{E}|$, where $|\mathcal{E}|$ is the cardinality of the set of edges $\mathcal{E}$ of $\mathcal{G}$, and informally, higher values of *arc strength* $\theta_{0,i}$ lead to the edge being traversed more often in the specified direction.

In this work, agents calculate the degree of belief (*i.e.*, a probability) of moving to a vertex v_i, given the *arc strengths*, by applying Bayes rule:

$$P(move_i|\theta_{0,i}) = \frac{P(move_i)P(\theta_{0,i}|move_i)}{P(\theta_{0,i})}. \tag{6}$$

The posterior probability $P(move_i|\theta_{0,i})$ is estimated via Bayesian inference from the prior $P(move_i)$ and likelihood $P(\theta_{0,i}|move_i)$ distributions. The denominator term is regarded as a normalization factor [15], being often omitted for the sake of simplicity.

The prior represents the belief obtained from analyzing past data. Naturally, in the MRPP, prior information about each vertex is encoded in the average idleness $\overline{\mathcal{I}_{v_i}}$ of a vertex v_i given by (2). Therefore, $P(move_i)$ is defined as:

$$P(move_i) = \frac{\overline{\mathcal{I}_{v_i}}}{\sum_{k=1}^{|\mathcal{V}|} \overline{\mathcal{I}_{v_k}}}, \tag{7}$$

thus decisions of moving to vertices with higher values of average idleness have intuitively higher probability. During the patrol mission, robots are continuously visiting new places and the $\overline{\mathcal{I}_V}$ values change over time. Each agent computes these values internally by tracking its own visits to $\mathcal{V}$ and communicating to other teammates when they arrive to a new vertex. In order to make an informed decision, at each decision step, the agent updates the prior information through (7), just before adopting (6) to obtain a degree of belief of moving to a neighbor vertex v_i.

In addition to the prior distribution, it is also necessary to define the likelihood through a statistical distribution to model the *arc strength* $\theta_{0,i}$. In the patrolling problem, agents must visit all $v_i \in \mathcal{G}$, thus, theoretically, assigning a uniform value for every arc would not be unreasonable. However, in such a dynamic system, where the number of visits to different locations in the environment is permanently evolving, it is usually advantageous to avoid traversing certain edges at a given time and favoring the use of others, in order to improve performance. Furthermore, task effectiveness is strongly related to the environment topology.

Hence, in the next subsection, a reward-based learning strategy to model and continually update the likelihood distribution is proposed in order to adapt to the system's state according to previous decisions, having a high impact on the behavior of robots and aiming at optimizing the collective performance.

B. Multi-Agent Reward-Based Learning

In general, reward-based learning methods are attractive since agents are programmed through reward and punishments without explicitly specifying how the task is to be achieved [16]. In this work, Bayesian Learning is employed to estimate the likelihood functions. Being a cooperative multi-robot task with lack of centralized control, with decentralized and distributed information and asynchronous computation, multiple simultaneous learners (one per patrolling agent) are involved.

The concept of delayed reward with a 1-step horizon model is explored. Each agent chooses an action of moving from v_0 to a neighbor v_i, based on (6). After reaching v_i, the information on its neighborhood has changed, namely the instantaneous idlenesses have been updated, *i.e.*, $\mathcal{I}_{v_i}(t) = 0$ and $\mathcal{I}_{v_0}(t) > 0$. Through information observed after making the move, a reward-based mechanism is used to punish or benefit the arcs involved in the decision to move from v_0 to v_i. This influences future moves starting in v_0, by introducing a bias towards arcs which ought to be visited ahead in time.

Henceforth, the reward-based learning method is explained. When the robot decides which one of the β neighbor vertices of v_0 is going to be visited next, each neighbor v_i will have an associated degree of belief given by the posterior probability. Therefore, it is possible to calculate the entropy:

$$H(move_i|\boldsymbol{\theta}) = -\sum_{i=1}^{\beta} P(move_i|\theta_{0,i})\ \log_2(P(move_i|\theta_{0,i})), \tag{8}$$

which measures the degree of uncertainty involved in the decision taken, being chosen for this reason as the basis for the punish/reward mechanism. The confidence on the decision taken is inversely proportional to the entropy H. Therefore, larger rewards and penalties are assigned to decisions with higher confidence (lower entropy). Note, however, that distinct v_i have different $deg(v_i)$ and, as a result, β varies for each decision instant. Therefore, the entropy is normalized to assume values in $[0, 1]$:

$$\mathcal{H}(move_i|\boldsymbol{\theta}) = \frac{H(move|\theta)}{\log_2(\beta)}. \tag{9}$$

After deciding and moving to a given v_k, the robot computes rewards for each arc between v_0 and its neighbor vertices v_i (including v_k) involved in the previous decision using:

$$\gamma_{0,i} = S_{0,i}(C_i, \mathcal{I}_{v_i}(t)) \cdot (1 - \mathcal{H}(move|\theta)), \tag{10}$$

with:

$$S_{0,i} \in \{-1, 0, 1\}. \tag{11}$$

$S_{0,i}$ gives the reward sign, providing a quality assessment which determines whether a penalty ($S = -1$), a reward ($S = 1$) or a neutral reward ($S = 0$) should be given. As can be seen, this function uses up-to-date information,

namely the number of visits to v_i, given by C_i, and the current instantaneous idleness $\mathcal{I}_{v_i}(t)$.

Prior to describing how $S_{0,i}$ is obtained, the normalized number of visits ζ_i is defined in (12). Note also that $N_G(v_0)$ represents the open neighborhood of v_0, *i.e.*, the set of adjacent vertices of v_0.

$$\zeta_i = \frac{C_i}{\deg(v_i)}, \tag{12}$$

$$\beta = \deg(v_0) = |N_G(v_0)|. \tag{13}$$

The sign of S is obtained using the set of rules defined below, which are checked as soon as the agent reaches v_i:

$$S_{0,i} = \begin{cases} -1, & \textbf{if } (\beta > 1) \wedge (\underset{j \in N_G(v_0)}{\operatorname{argmax}}\ \zeta_j = i) \wedge \\ & (|\underset{j \in N_G(v_0)}{\operatorname{argmax}}\ \zeta_j| = 1); \\ -1, & \textbf{if } (\beta > 1) \wedge (\underset{j \in N_G(v_0)}{\operatorname{argmax}}\ \zeta_j = i) \wedge \\ & (|\underset{j \in N_G(v_0)}{\operatorname{argmax}}\ \zeta_j| > 1) \wedge (\underset{j \in N_G(v_0)}{\operatorname{argmin}}\ \mathcal{I}_{v_j}(t)) = i); \\ 1, & \textbf{if } (\beta > 1) \wedge (\underset{j \in N_G(v_0)}{\operatorname{argmin}}\ \zeta_j = i) \wedge \\ & (|\underset{j \in N_G(v_0)}{\operatorname{argmin}}\ \zeta_j| = 1); \\ 1, & \textbf{if } (\beta > 1) \wedge (\underset{j \in N_G(v_0)}{\operatorname{argmin}}\ \zeta_j = i) \wedge \\ & (|\underset{j \in N_G(v_0)}{\operatorname{argmin}}\ \zeta_j| > 1) \wedge (\underset{j \in N_G(v_0)}{\operatorname{argmax}}\ \mathcal{I}_{v_j}(t)) = i); \\ 0, & \textbf{otherwise}. \end{cases} \tag{14}$$

ζ_i is used in the punish/reward procedure because higher degree vertices are naturally more visited than vertices with lower degree, being often traversed to reach isolated vertices that tend to have a lower number of visits. Note also that $\underset{j \in N_G(v_0)}{\operatorname{argmax}}\ \zeta_j$ and $\underset{j \in N_G(v_0)}{\operatorname{argmin}}\ \zeta_j$ may return more than one solution, therefore the cardinality of the solution set, $|\underset{j \in N_G(v_0)}{\operatorname{argmax}}\ \zeta_j|$ and $|\underset{j \in N_G(v_0)}{\operatorname{argmin}}\ \zeta_j|$, is checked to ensure that strictly one reward and one punishment are assigned. As such, the assignment of $S_{0,i}$ respects the following criteria:

- $S_{0,i} = -1$, when the degree of v_0 is higher than one ($\beta > 1$) and the normalized number of visits to v_i (ζ_i) is maximal in the neighborhood of v_0. In case there is more than one vertex with maximal ζ, a negative reward is given to the one with lower instantaneous idleness $\mathcal{I}_{v_j}(t)$ between those.
- $S_{0,i} = 1$, when the degree of v_0 is higher than one ($\beta > 1$) and ζ_i is minimal in the neighborhood of v_0. In case there is more than one vertex with minimal ζ, a positive reward is given to the one with higher instantaneous idleness $\mathcal{I}_{v_j}(t)$ between those.
- $S_{0,i} = 0$, in every other situation.

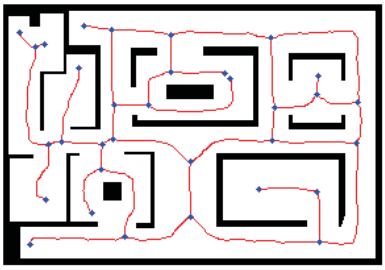


Fig. 2: Environment used in the experiments with respective topological map.

In the beginning of the mission, when $t = t_0$, all *arcs strength* $\theta_{0,i}$ are equal to a real positive number κ:

$$\forall \theta_{0,i} \in \theta,\ \theta_{0,i}(t_0) = \kappa. \tag{15}$$

As the mission evolves, the agent updates $\theta_{0,i}$ through:

$$\theta_{0,i}(t) = \theta_{0,i}(t-1) + \gamma_{0,i}(t). \tag{16}$$

Note that the larger the value of κ is set in (15), the less immediate influence the rewards received will have on $\theta_{0,i}$. In the experimental tests, $\kappa = 1.0$ was used. This reward-based procedure is expected to make the values of $\theta_{0,i}$ fluctuate as time goes by, informing robots of moves which are potentially more effective, but keeping in mind that robots must visit all vertices v_i in the patrolling mission.

Finally, the learnt likelihood distribution is obtained through normalization of $\theta_{0,i}$:

$$P(\theta_{0,i}|move_i) = \frac{\theta_{0,i}}{\sum_{j=1}^{|\mathcal{E}|}\sum_{k=1}^{|\mathcal{E}|}\theta_{j,k}}, \tag{17}$$

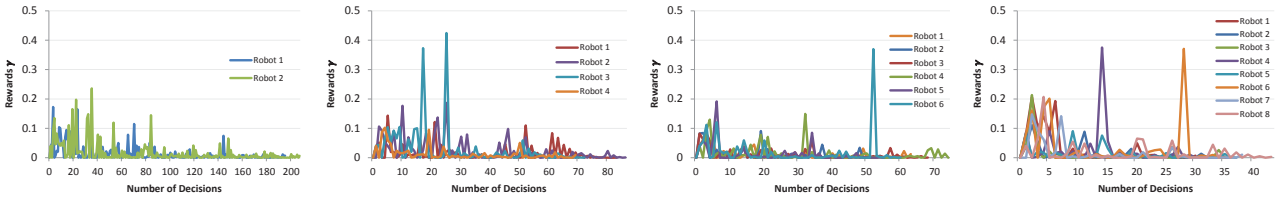
being updated at each decision step and making use of experience acquired in the past for future decisions.

C. Decision-Making and Multi-Agent Coordination

In CBLS each independent agent is adapting its behavior via its own learning process and has no control or knowledge of how other agents behave nor their internal state, *i.e.*, they do not know their teammates' likelihood distribution $P(\theta_{0,i}|move_i)$ and cannot predict their moves.

In collective operations with a common objective, coordination among agents plays a fundamental role in the success of the mission. Particularly in this context, it is highly undesirable that agents move to the same locations. Therefore, an asynchronous and distributed communication system is used to inform teammates of an agent's current vertex v_0, as well as the vertex v_i chosen for its next move.

By simply exchanging this messages with its teammates, each robot can update the information about the state of the system, namely the idleness values, and decide its moves taking that information into account, as well as its progressively acquired experience. Local coordination arises by inspecting if another robot has expressed intention to move to a given vertex v_i in the local neighborhood and if so, remove v_i from the decision. Finally, the decision-making process of the



(a) 2 robots. (b) 4 robots. (c) 6 robots. (d) 8 robots.

Fig. 3: Evolution of the absolute reward values along four experiments with different teamsize.

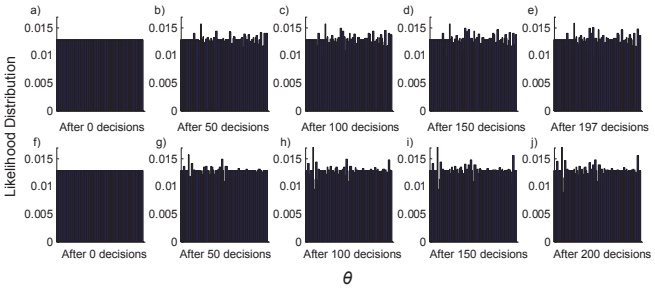


Fig. 4: Evolution of the likelihood distribution in a mission with 2 robots. a) to e) different decisions instants for robot 1; f) to j) different decisions instants for robot 2.

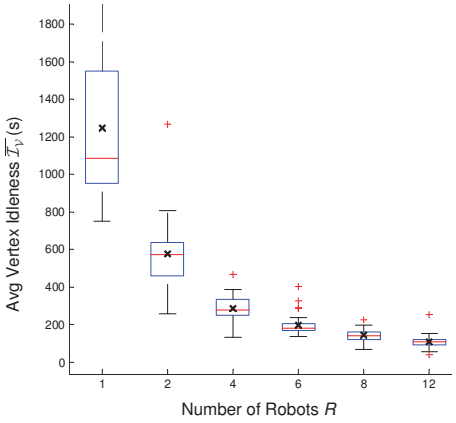


Fig. 5: Overall results running CBLS with different teamsize.

agent consists of choosing the move from v_0 to the neighbor vertex v_j with the maximum probability among all possible decisions:

$$move_j = \text{true}: \quad j = \underset{i \in N_{\mathcal{G}}(v_0)}{\text{argmax}}\, P(move_i|\theta_{0,i}) \tag{18}$$

IV. Results and Discussion

In order to assess the performance of CBLS, a set of simulation experiments have been conducted. To that end, the environment illustrated in Fig. 2 has been used to test the approach with different teamsizes of $R = \{1, 2, 4, 6, 8, 12\}$ robots. A recognized simulator with realistic modeling was chosen: the Stage 2D multi-robot simulator, while ROS was adopted to program the robots.

The graph information of the environment is loaded by every robot in the beginning of each simulation, which then runs the described algorithm. Robots navigate safely in the environment by heading towards their goals while avoiding collisions through the use of ROS navigation stack and an adaptive Monte Carlo localization approach. Additionally, robots have non-holonomic constraints and travel at a maximum velocity of 0.2 m/s. All the simulations conducted respect a stopping condition determined by 4 complete patrolling cycles i.e., after every $v_i \in \mathcal{G}$ has been visited at least 4 times. This stopping condition is adequate, because $\overline{\mathcal{I}_{\mathcal{G}}}$ converged in all experiments conducted.

It can be seen in the histograms of Fig. 4 the evolution of the likelihood function in the example of a patrolling mission with two robots. Note that each robot apprehends a different distribution and has no control or knowledge on the internal state of its teammate. As expected, peaks in the histograms emerge with the increasing number of decisions. Despite that, it is also clear that values fluctuate around the initial uniform value, which comes as a consequence of robots having to visit every vertex $v_i \in \mathcal{G}$.

Another interesting aspect observed in the experiments is the descending trend shown by the absolute reward values along different experiments, which are given by the $(1 - \mathcal{H})$ factor in (10). Fig. 3 illustrates how these values evolve in missions with four different teamsizes. Despite the occasional peaks that occur with larger number of robots, such values tend to decrease with the number of decisions. This is because, in general, as the system progresses, the $\overline{\mathcal{I}_{\mathcal{V}}}$ values of different vertices become more balanced and, as a result, the degree of belief in moving to distinct neighbors comes closer. In such situations, the closer the posterior probabilities are, the higher the entropy becomes, therefore the reward values descend gradually. The peaks observed are justified by situations where agents share nearby areas, temporarily perturbing the $\overline{\mathcal{I}_{\mathcal{V}}}$ values in the neighborhood of other agents. For that reason, peaks are more observable in larger teams.

Moving on to the performance of the algorithm, the boxplot chart in Fig. 5 represents the $\overline{\mathcal{I}_{\mathcal{V}}}$ values (in seconds) for each tested teamsize. The average value is represented by a black cross, providing a generalized measure: the average graph idleness, $\overline{\mathcal{I}_{\mathcal{G}}}$ (*cf.* Eq. 3). The ends of the blue boxes and the horizontal red line in between correspond to the first and third quartiles and the median values of $\overline{\mathcal{I}_{\mathcal{V}}}$, respectively.

As expected, the idleness values decrease when the number of robots grow. Despite the increasing performance displayed by the CBLS approach, the individual contribution of adding more robots gradually reduces with teamsize. Group productivity will eventually converge with a large

TABLE I: Final $\overline{\mathcal{I}_\mathcal{G}}$ values (in seconds) using different strategies and the map of Fig.2.

Teamsize	CR	HCR	HPCC	CGG	MSP	GBS	SEBS	**CBLS**
1	1315.79	1283.59	1235.67	1347.30	1401.80	1267.26	1277.16	**1249.45**
2	675.44	654.61	670.44	675.64	749.42	708.82	671.18	**575.06**
4	363.46	373.45	298.77	335.45	375.15	351.19	339.93	**284.88**
6	238.57	273.60	254.96	234.18	248.92	275.98	230.39	**197.33**
8	198.90	217.38	225.44	172.39	185.28	206.19	197.03	**143.36**
12	172.40	255.62	212.30	143.94	-	145.89	118.73	**108.22**

R. In theory, productivity should grow during size scale-up; however, spatial limitations increase the number of times the robots meet and beyond a given R, it is argued that they will spend more time avoiding each other than effectively patrolling on their own.

Another interesting aspect illustrated in the boxplot of Fig. 5 is that the median $\widetilde{\mathcal{I}_\mathcal{G}}$ is lower than the mean ($\overline{\mathcal{I}_\mathcal{G}}$) in all configurations. This means that the $\mathcal{I}_\mathcal{V}$ values are positively skewed, *i.e.*, most of the values are below the average, $\overline{\mathcal{I}_\mathcal{G}}$.

Using findings from previous works (*cf.* [12], [14]), where tests in the same map and benchmarking with several state-of-the-art patrolling approaches were conducted, Table I was built. In this table, performance of 8 approaches, including CBLS, with the same teamsizes is compared using the $\overline{\mathcal{I}_\mathcal{G}}$ metric. The results obtained in this work are depicted in the last column, and as can be seen, these clearly outperform the rest of the approaches. For more details on the various strategies and tests previously conducted, the interested reader should refer to [12] and [14].

Finally, on a general note, visual inspection of the trajectories of robots using CBLS showed that prediction of patrolling routes is far from being straightforward, as opposed to most strategies presented in Table I. This stochastic behavior, together with the promising results obtained, proves the effectiveness of the approach and the potential to apply it in actual security systems with physical teams of robots.

V. Conclusion

In this work, cooperative multi-agent learning has been addressed in order to solve the patrolling problem in a distributed way. Robots make use of Bayesian decision to reason on their moves so as to patrol effectively an environment, while coordinating their behaviors. Concurrent reward-based learning has been adopted given that, in this domain, the decomposition of the problem reduces the complexity of the general cooperative mission by distributing computational load among each independent learner.

Experimental results have shown that the method is able to tackle the problem, since it can deal with uncertainty and the actions are selected according not only to prior knowledge about the problem, but also the state of the system at the time, resulting in adaptive, effective and distributed cooperative patrolling. Additionally, when placed in comparison with several state-of-the-art approaches, CBLS outperforms them independently of teamsize.

In the future, beyond testing the approach with physical robots in real environments, it might be interesting to relax the assumption of perfect communication, testing the performance using only local interactions between robots in the same range, similarly to our previous work [14]. Finally, adding to the decision knowledge from beyond the local neighborhood of a robot could potentially improve the performance even further.

Acknowledgment

The authors gratefully acknowledge Luís Santos (ISR, University of Coimbra) for his contribution and feedback.

References

[1] F. Sempé and A. Drogoul, "Adaptive Patrol for a Group of Robots". In Proc. *Int. Conf. on Int. Robots and Sys. (IROS'03)*, Las Vegas, 2003.

[2] Y. Chevaleyre, "Theoretical analysis of the multi-agent patrolling problem". In Proc. of the *Int. Conf. on Agent Intelligent Technologies (IAT'04)*, Beijing, China, September 20-24, pp. 30–308, 2004.

[3] C. Pippin, H. Christensen and L. Weiss, "Performance Based Task Assignment in Multi-Robot Patrolling". In Proc. of *ACM Symp. on Applied Computing (SAC '13)*, Coimbra, Portugal, March 18-22, 2013.

[4] N. Basilico, N. Gatti, T. Rossi, S. Ceppi and F. Amigoni, "Extending Algorithms for Mobile Robot Patrolling in the Presence of Adversaries to More Realistic Settings". In Proc. of the *Int. Conf. on Intelligent Agent Tech. (IAT'09)*, pp. 557-564, Milan, Italy, 2009.

[5] J. Marier, C. Besse and B. Chaib-draa, "Solving the Continuous Time Multiagent Patrol Problem". In Proc. of the *Int. Conf. on Robotics and Automation (ICRA'10)*, Anchorage, Alaska, USA, May, 2010.

[6] P. Sampaio, G. Ramalho and P. Tedesco, "The Gravitational Strategy for the Timed Patrolling". In Proc. of the *Int. Conf. on Tools with Artificial Intelligence (ICTAI'10)*, Arras, France, Oct. 27-29, 2010.

[7] F. Pasqualetti, A. Franchi and F. Bullo, "On cooperative patrolling: optimal trajectories, complexity analysis, and approximation algorithms". In *IEEE Transactions on Robotics*, 28 (3), pp. 592-606, June, 2012.

[8] Y. Ishiwaka, T. Sato and Y. Kakazu, "An approach to the pursuit problem on a heterogeneous multiagent system using reinforcement learning". In *Robotics and Autonomous Systems (RAS)*, Elsevier, 43 (4), June 2003.

[9] H. Santana, G. Ramalho, V. Corruble and B. Ratitch, "Multi-Agent Patrolling with Reinforcement Learning". In Proc. of the *Int. Conf. on Aut. Agents and Multiagent Sys.*, Vol. 3, New York, 2004.

[10] X. Chen and T.S. Yum, "Patrol Districting and Routing with Security Level Functions". In Proc. of the *Int. Conf. on Systems, Man and Cybernetics (SMC'2010)*, pp. 3555-3562, Istanbul, Turkey, Oct. 2010.

[11] L. Iocchi, L. Marchetti and D. Nardi, "Multi-Robot Patrolling with Coordinated Behaviours in Realistic Environments". In Proc. of the *Int. Conf. on Intelligent Robots and Systems (IROS'2011)*, pp. 2796-2801, San Francisco, CA, USA, September 25-30, 2011.

[12] D. Portugal and R.P. Rocha, "Multi-Robot Patrolling Algorithms: Examining Performance and Scalability". In *Advanced Robotics Journal*, 27 (5), pp. 325-336, March, 2013.

[13] D. Portugal and R.P. Rocha, "Decision Methods for Distributed Multi-Robot Patrol". In Proc. of the *Int. Symposium on Safety, Security and Rescue Robotics (SSRR'2012)*, College Station, TX, USA, Nov. 2012.

[14] D. Portugal and R.P. Rocha, "Distributed Multi-Robot Patrol: A Scalable and Fault-Tolerant Framework". In *Robotics and Autonomous Systems (RAS)* Journal, Elsevier, 2013. (In Press)

[15] F. Jansen and T. Nielsen, "Bayesian Networks and Decision Graphs", 2nd Edition, Springer Verlag, 2007.

[16] L. Panait and S. Luke, "Cooperative Multi-Agent Learning: The State of the Art", In *Journal of Autonomous Agents and Multi-Agent Systems*, 11(3), pp. 387-434, November 2005.