Chapter 16
Cooperative Multi-robot Patrol in an Indoor Infrastructure

David Portugal and Rui P. Rocha

Abstract Multi-robot patrol (MRP) is essentially a collective decision-making problem, where mobile robotic units must coordinate their actions effectively in order to schedule visits to every critical point of the environment. The problem is commonly addressed using centralized planners with global knowledge and/or calculating a priori routes for all robots before the beginning of the mission. However, distributed strategies for MRP have very interesting advantages, such as allowing the team to adapt to changes in the system, the possibility to add or remove patrol units during the mission, and leading to trajectories that are much harder to predict by an external observer. In this work, we present a distributed strategy to solve the patrolling problem in a real world indoor environment, where each autonomous agent decides its actions locally and adapts to the system's needs using distributed communication. Experimental results show the ability of the team to coordinate so as to visit every important point of the environment. Furthermore, the approach is able to scale to an arbitrary number of robots as well as overcome communication failures and robot faults.

16.1 Introduction

This work addresses multi-robot systems (MRS) for cooperative patrolling missions in realistic scenarios. To patrol is herein defined as "the activity of going around or through an area at regular intervals for security purposes" [1]. It requires every position in the environment, or at least the ones that need surveillance, to be regularly visited to verify the absence of anomalies. Additionally, it aims at monitoring

D. Portugal (✉) · R.P. Rocha
Institute of Systems and Robotics (ISR), University of Coimbra (UC),
3030-290 Coimbra, Portugal
e-mail: davidbsp@isr.uc.pt

R.P. Rocha
e-mail: rprocha@isr.uc.pt

P. Spagnolo et al. (eds.), *Human Behavior Understanding in Networked Sensing*,
DOI 10.1007/978-3-319-10807-0_16

environments, obtaining information, searching for objects, and clearing areas in order to guard the grounds from intrusion. Consequently, performing a patrolling mission with a team of any given number of autonomous and cooperative robots distributed in space represents a complex challenge. Being monotonous and repetitive, patrolling missions may also be dangerous (e.g., patrolling in hazardous environments). Therefore, using MRS in this context enables to safeguard human lives in applications like mine clearing, search and rescue operations, or surveillance, reducing the risk to human operators, which can be occupied in nobler tasks like monitoring the system from a safer location [2, 3].

In Multi-robot patrol (MRP), it is common to abstract the environment through a topological, graph-like map and robots are expected to have improved sensing abilities, meaning that they need to visit regularly all important places in the environment without necessarily going everywhere. Thus, agents should coordinate their actions in a shared environment while continuously deciding which place to move next after clearing their locations, having the ultimate goal of achieving optimal group performance.

The problem is commonly addressed using centralized planners with global knowledge and/or calculating a priori routes for all robots before the beginning of the mission. However, the deterministic nature of such methods do not capture the repetitive, and hence dynamic aspect of the problem, nor the synchronization issues that arise when a timing among the visits of certain zones is required. On the other hand, distributed strategies for MRP have very interesting advantages, such as allowing the team to adapt to changes in the system, the possibility to add or remove patrol units during the mission, and leading to trajectories that are much harder to predict by an external observer.

In this work, a distributed algorithm is proposed to solve the problem. The patrolling route of each robot is progressively built online and according to the state of the system. Furthermore, all robots are endowed with autonomous decision-making capabilities, being able to decide their moves by resorting to Bayesian decision, instead of following routes computed by a centralized entity.

In the next section, a literature review is conducted and the contributions of the chapter are described. Afterward, the MRP Problem is defined and the performance metric is presented in Sect. 16.3. The following section describes the distributed multi-robot patrolling strategy adopted in this chapter. Section 16.5 presents the experimental setup and the results obtained in real world indoor patrolling missions, as well as a discussion of the facets of the problem. Finally, the chapter ends with conclusions and open issues for future research.

16.2 Related Work

An overview of multi-robot strategies for area patrol missions is presented in this section. The MRP problem is also known in the literature as repetitive sweeping, multi-robot monitoring, and graph coverage, and the interest in this field stems from the variety of possible approaches, the potential applications of such algorithms in several distinct areas, and the key social function of these systems.

Important theoretical contributions to the MRP have been presented by [4–6]. Considering the *idleness* criterion (cf. Sect. 16.3), it was shown that the problem is NP-Hard, i.e., no polynomial time algorithm is known to compute an optimal solution to the problem. Also, it was shown that in theory, it can be optimally solved with a single robot by finding a Traveling Salesman Problem (TSP) tour in the graph that describes the environment. However, such computation is also NP-Hard. For the multi-robot case the problem remains open with several different algorithms in terms of motion rules, communication paradigm, cooperation scheme, etc. being proposed by several research groups in the last decade.

Numerous works employ graph theory tools like spanning trees [7] or graph partitioning [8] to compute minimal-cost cycles that assign efficient routes for each robot in the patrolling mission. Auctions and market-based coordination are also popular among MRP literature as in the case of [9], where agents bid to exchange vertices of the patrol graph to increase overall patrol performance. Diverse other concepts have also been explored, such as simple reactive architectures [10], task allocation [11], artificial forces [12], Gaussian process theory [13], evolutionary algorithms [14], swarm intelligence [15], Markov decision process [16], linear programming modeling [17], and others. A survey of methods in the literature for MRP can be found in [18].

Research on adversarial patrolling, a variant of the classical multi-robot area patrol, has addressed important issues such as applying unpredictable actions in the patrolling method so that intruders will not have access to the patrolling trajectory information to avoid being detected by agents [19], and studying strategies to detect intruders with high probability [20]. In identical surveillance application scenarios, where the threat of having adversarial agents is a central issue, the problem can also be addressed using game-theoretic approaches [21, 22].

Most contributions to the literature propose patrolling methods and overlook other relevant problems that should be addressed in MRP missions, e.g., robustness to failures and scalability. Furthermore, the vast majority of related work verifies their results only in simulations with different types of assumptions; some exceptions being the works described in [9, 23] and a few others. Clearly, there is a lack of implementation using physical MRS in noncentralized architectures. This serves as a motivation for the need to continuously coordinate teams of robots in patrolling missions in a distributed way, to validate these systems in the real-world and the possibility to add or remove patrolling agents (e.g., due to failures).

As will be seen, the main advantages of the patrolling framework presented in this chapter are the adaptability to the system's needs, which results from providing the robots with autonomous decision-making capability; the straightforwardness of implementation; and the quality of the technique leading to a scalable and fault-tolerant solution, which represents the main contribution of this work to the state of the art.

16.3 Problem Definition

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 map of the environment and through a graph extraction algorithm [24], they obtain an undirected, connected and metric navigation graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $v_i \in \mathcal{V}$ vertices and $e_{ij} \in \mathcal{E}$ edges. Each vertex represents a specific location that must be visited regularly and each edge represents the connectivity between these locations, having a weight $|e_{i,j}|$ defined by the metric distance between v_i and v_j. $|\mathcal{V}|$ represents the cardinality of the set $\mathcal{V}$ and $|\mathcal{E}|$ represents the cardinality of the set $\mathcal{E}$. Seeing as undirected graphs are assumed, then: $|\mathcal{E}| \leq \frac{|\mathcal{V}|\cdot(|\mathcal{V}|-1)}{2}$.

Informally, a good strategy is one that minimizes the time lag between two passages to the same place and for all places [4]. Thus, the MRP problem can be reduced to coordinate robots in order to frequently visit all vertices of the graph, ensuring the absence of atypical situations with respect to a predefined optimization criterion.

In order to address and compare the performance of different patrolling algorithms, it is important to establish an evaluation metric. Diverse criteria 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 [23]. In this work, the first one has been considered [25, 26], given that it measures the elapsed time since the last visit from any agent in the team to a specific location. Idleness is intuitive to analyze and brought into confrontation with the possibility of attacks to the system, seen as it uses time units. Thus, in the following equations, we define important variables used in the upcoming sections of the chapter.

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

$$\mathcal{I}_{v_i}(t) = t - t_l, \tag{16.1}$$

wherein 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:

$$\overline{\mathcal{I}_{v_i}}(t) = \frac{\overline{\mathcal{I}_{v_i}}(t_l) \cdot C_i + \mathcal{I}_{v_i}(t)}{C_i + 1}, \tag{16.2}$$

where C_i represents the number of visits to v_i. Considering now $\overline{\mathcal{I}_{\mathcal{V}}}$ as the set of the average idlenesses of all $v_i \in \mathcal{V}$, given by:

$$\overline{\mathcal{I}_{\mathcal{V}}} = \{\overline{\mathcal{I}_{v_1}}, ..., \overline{\mathcal{I}_{v_i}}, ..., \overline{\mathcal{I}_{v_{|\mathcal{V}|}}}\}, \tag{16.3}$$

the maximum average idleness of all vertices $\max(\overline{\mathcal{I}_{\mathcal{V}}})$ in time step t is defined as:

$$\max(\overline{\mathcal{I}_{\mathcal{V}}})(t) = \max\{\overline{\mathcal{I}_{v_1}}(t), ..., \overline{\mathcal{I}_{v_i}}(t), ..., \overline{\mathcal{I}_{v_{|\mathcal{V}|}}}(t)\}. \tag{16.4}$$

For the sake of simplicity, the argument (t) is omitted whenever timing is not relevant. Finally, in order to obtain a generalized measure, the average idleness of the graph $\mathcal{G}$ ($\overline{\mathcal{I}_{\mathcal{G}}}$) is defined as:

$$\overline{\mathcal{I}_{\mathcal{G}}} = \frac{1}{|\mathcal{V}|} \sum_{i=1}^{|\mathcal{V}|} \overline{\mathcal{I}_{v_i}}. \tag{16.5}$$

A similar assumption to other works in the literature [4, 10] is taken in the beginning of the experiments, where for all $v_i \in \mathcal{V}$, $\mathcal{I}_{v_i}(0) = 0$, as if every vertex had just been visited, when the mission started. As a consequence, 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, as will be seen in the results section. For this reason, the final $\overline{\mathcal{I}_{\mathcal{G}}}$ value is measured only after convergence in the stable phase.

Considering a patrol path as an array of vertices of $\mathcal{G}$, the multi-robot patrolling problem can be described as the problem of finding a set of R robot trajectories $\mathbf{x}$ that visit all vertices $v_i \in \mathcal{V}$ of the graph $\mathcal{G}$, using an arbitrary team of R robots, with the overall team goal of minimizing $\overline{\mathcal{I}_{\mathcal{G}}}$:

$$f = \underset{\mathbf{x}}{\operatorname{argmin}}(\overline{\mathcal{I}_{\mathcal{G}}}), \tag{16.6}$$

by finding:

$$\mathbf{x} = \{x_1, ..., x_r, ..., x_R\}, \tag{16.7}$$

such that:

$$x_r = \{v_a, v_b, ...\}, \tag{16.8}$$

$$v_a, v_b, ... \in \mathcal{V},$$

$$1 \leq r \leq R, R \in \mathbb{N},$$

subject to:

$$\forall v_i \in \mathcal{V}, \exists x_r \in \mathbf{x} : v_i \in x_r. \tag{16.9}$$

Note that x_r represents the patrolling path of robot r, which has an arbitrary dimension that depends on each robot's decisions and v_a, v_b, ... are generic vertices in $\mathcal{V}$, which do not imply any specific order.

In this work, instead of relying in precomputed routes, which is common in classical approaches, the patrolling route x_r of each robot is built online according to the state of the system. Furthermore, all robots are able to decide their own moves instead of following routes that are computed by a centralized coordinator.

16.4 A Distributed Strategy for MRP

In [27], it was shown that methods based on Bayesian principles can effectively solve the MRP problem in different environment topologies. These methods can cope with uncertainty and robots' actions are selected according to the state of the system, leading to adaptive and distributed cooperative patrolling. More specifically, in the State-Exchange Bayesian Strategy (SEBS) each robot takes other teammates' actions into account while, at the same time, aiming to maximize a local gain function, which guarantees that every vertex of the navigation graph is visited regularly by all agents.

In order to compare the performance of diverse state of the art MRP techniques, we have used Stage [28], a recognized multi-robot simulator designed to support research into multi-agent autonomous systems. Stage simulates not only a population of mobile robots, but also sensors and objects in a two-dimensional environment. Moreover, it is realistic for many purposes, having a well-known status of being a robust simulation platform. In order to program the robots, the robot operating system (ROS) [29] was adopted. It was also verified that the nodes developed using Stage would work with little or no modification with real robots.

In the comparative analysis conducted, SEBS outperformed the remaining methods [27]. As a consequence, in this work it is chosen as the preferred strategy for coordination of a team of mobile robots, in a patrolling mission in a real world indoor infrastructure.

In the following subsections, we review the SEBS strategy. Using this strategy, agents decide asynchronously which place to move next when they reach their location, according to prior information about the team and the environment. Having this in mind, a fundamental random variable, which represents the act of moving (or not) to a neighbor vertex v_A is defined as:

$$\text{move}(v_A) = \{\text{true, false}\}, \tag{16.10}$$

The variables, which influence each robot's decision to move to a specific vertex, are presented in the next subsections with special focus on the selection of proper statistical distributions to model the data, so as to ensure the quality of the results. Afterward, it is shown how the local decision-making process is automated, applying Bayes Rule.

16.4.1 Gain

After reaching a given location of the environment, which needs to be visited, each robot is faced with a decision, where it must decide the next move within the shared environment. Therefore, the Gain G_A of moving from the current vertex (v_0) to a neighbor vertex (v_A), assuming constant speed (c) is defined as:

$$G_A(t) = c \cdot \left(\frac{\mathcal{I}_{v_A}(t) - \mathcal{I}_{v_A}(t + \Delta t)}{|e_{\text{val}}|} \right), \tag{16.11}$$

where in $t + \Delta t$ is the arrival time in v_A, and $\Delta t = {}^{|e_{0A}|}/_{c}$. $G_A(t)$ is proportional to a difference in the idleness values, representing a gain that the robot expects to obtain in moving to a given vertex. Note however that $G_A(t) \geq 0$ because $\mathcal{I}_{v_A}(t + \Delta t) = 0$ when the robot reaches v_A. For simplicity of notation, let us omit (t) in G_A, since every computation is done instantaneously.

In most cases, $|e_{\text{val}}|$ is equal to $|e_{0A}|$, which represents the distance between v_0 and v_A, given by the weight of the edge that connects them. However, the constraint (16.12) is imposed in order to control the impact of $|e_{\text{val}}|$, avoiding seldom situations where robots may get trapped in local optima (i.e., repeatedly visiting vertices that are very close to each other):

$$|e_{\text{val}}| = \begin{cases} |e_{\min}|, \text{ if } \max\{e_{0A}, ..., e_{0\beta}\} > 2\min\{e_{0A}, ..., e_{0\beta}\} & \wedge \quad |e_{0A}| < |e_{\min}|; \\ |e_{0A}|, \text{ otherwise.} \end{cases} \tag{16.12}$$

Naturally, greater values of gain rapidly have more influence in the robot's decision. Hence, the distribution function $F(g)$ of G_i is defined as a monotonically increasing function with the exponential model, as illustrated in Fig. 16.1a:

$$F(g) = ae^{bg}; \quad a > 0, \tag{16.13}$$

where:

$$F(0) = L \Leftrightarrow a = L, \tag{16.14}$$

and:

$$1 = Le^{bM} \Leftrightarrow b = \frac{\ln({}^{1}/_{L})}{M}. \tag{16.15}$$

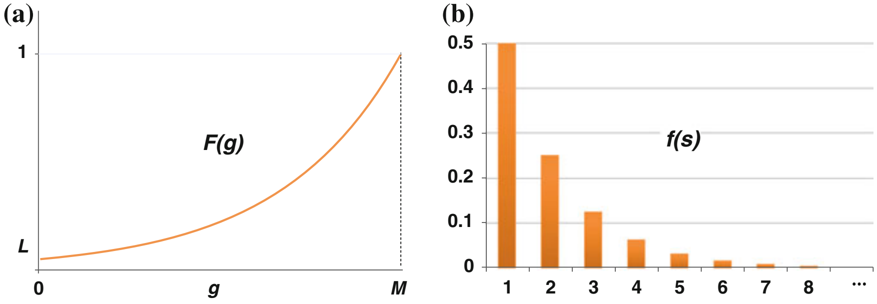

Fig. 16.1 Distribution Functions. **a** Gain, $F(g)$, **b** State, $f(s)$

This results in:

$$F(g) = L \cdot \exp\left(\frac{\ln\left({}^{1}\!/\!{}_{L}\right)}{M} g\right), \tag{16.16}$$

with:

$$L, M > 0 \quad \text{and} \quad g < M. \tag{16.17}$$

L and M are constants that control the distribution function. Particularly, L is the y-intercept, which controls the probability values for lower gains and M is the gain saturation, beyond which the probability values are maximum; $F(g \geq M) = 1$. These constants are simply defined as $L = 0.1$ in the experiments conducted; and M is calculated through (16.11) using an upper bound of $\mathcal{I}_{v_A}$.

16.4.2 State

If the robots only depended on the Gain variable, they would aim to obtain the best reward for themselves, neglecting the global objective of the patrolling team and acting independently of their teammates, i.e., they would follow a greedy approach. However, in collective operations with a common goal, coordination between agents is fundamental for the success of the mission. Particularly in the MRP context, it is highly undesirable that agents move to the same positions. Instead, they should communicate their goals to ensure the distribution in the shared environment. As a result, a random variable, the vertex state S_i, is defined to model the number of robots that intend to visit a given vertex v_i involved in the decision process of robot r, which is currently located in vertex v_0:

$$S_i \in \mathbb{N}_0 \cap [0, R-1]; \quad R > 1. \tag{16.18}$$

As previously, it is necessary to define a statistical distribution $f(s)$ to model the vertex state. The greater the number of teammates in a given region in the vicinity of a robot, it becomes increasingly unlikely for the robot to move in that direction. To describe this behavior, the following probability mass function has been defined, which uses a geometric sequence of ratio 1⁄2:

$$f(s)_{R\to\infty} = P(S_i = s)_{R\to\infty} = \frac{1}{2^{s+1}}, \tag{16.19}$$

as shown in Fig. 16.1b. This geometric sequence is used to guarantee that the total probability for all S_i equals 1:

$$\sum_{s=0}^{R-1} f(s) = 1, \tag{16.20}$$

Equation (16.19) assumes that the number of robots R is unknown and can be arbitrarily high. Nevertheless, given that the robots communicate among themselves in order to coordinate, it is more realistic to consider R as known and with finite values. Thus, the following approximation to (16.19) is assumed:

$$f(s) = P(S_i = s) = \frac{2^{R-(s+1)}}{2^R - 1}; \quad R > 1. \tag{16.21}$$

16.4.3 Robot Decision

Each decision to move from a vertex v_0 to a neighbor v_A is independent and agents have the ability to choose the action, which has the greatest expectation of utility, weighted by the effects of all possible actions.

Having this in mind, robots locally update the idleness values online, by communicating to other robots when they reach any vertex of the navigation graph, in a distributed way. This enables the calculation of the Gain of moving from the current vertex to any of its neighbors. Similarly, by receiving other robots' intentions, agents can calculate the vertex state.

Having the likelihood distribution models $P(G_i|\text{move})$ and $P(S_i|\text{move})$ defined respectively by $F(g)$ and $f(s)$; and considering the prior knowledge as uniform, $P(\text{move})$, where all decisions are equiprobable; agents calculate the probability of moving to a specific vertex i, given its gain G_i and the vertex state S_i, by applying

Algorithm 1: State Exchange Bayesian Strategy (SEBS).

```
while true do
    add($v_n$ to $x_r$);   //current vertex
    forall $v_i \in N_{\mathcal{G}}(v_n)$ do
        $G_i \leftarrow c \cdot \left(\frac{\mathcal{I}_{v_i}(t) - \mathcal{I}_{v_i}(t+\Delta t)}{|e_{val}|}\right)$;
        $P(G_i|move(v_i)) \leftarrow L \cdot \exp\left(\frac{\ln(1/L)}{M} G_i\right)$;
        $S_i \leftarrow$ count_intentions_to($v_i$);
        $P(S_i|move(v_i)) \leftarrow \frac{2^{R-(S_i+1)}}{2^R-1}$;
        $P(move(v_i)|G_i, S_i) \propto P(move(v_i))P(G_i|move(v_i))P(S_i|move(v_i))$;
    //Next vertex is the neighbor of the current vertex with highest posterior probability.
    $v_{n+1} \leftarrow \operatorname{argmax}_{i \in N_{\mathcal{G}}(v_n)} P(move(v_i)|G_i, S_i)$
    write_msg_arrival_to($v_n$);
    write_msg_intention_to($v_{n+1}$);
    while move_robot to $v_{n+1}$ do
        read_msg_arrival_and_intentions_to($\mathcal{V}$);
        update($\mathcal{I}_{\mathcal{V}}(t)$);
    $v_n \leftarrow v_{n+1}$;
```

Bayes rule:

$$P(\text{move}(v_i)|G_i, S_i) = \frac{P(\text{move}(v_i))P(G_i|\text{move}(v_i))P(S_i|\text{move}(v_i))}{P(G_i)P(S_i)}. \tag{16.22}$$

The denominator term is a normalization factor [30], being omitted for simplification purposes. Finally, the decision-making process of the agent consists of choosing the move from v_0 to the vertex v_j with the maximum probability among all vertices of the neighborhood $N_{\mathcal{G}}(v_0)$ of v_0:

$$\text{move}_j = \mathsf{true} : \quad j = \underset{i \in N_{\mathcal{G}}(v_0)}{\mathtt{argmax}}\, P(\text{move}(v_i)|G_i, S_i) \tag{16.23}$$

Algorithm 1 presents a high-level pseudo-code of the SEBS approach, which runs locally on each mobile robot.

16.5 Results and Discussion

In this section, the implementation of a system for multi-robot patrol in a real-world environment is presented to demonstrate the potential of employing mobile robots as a potential solution in surveillance missions. Aiming to fill a gap in the state of the art, the SEBS distributed approach is validated in a large indoor scenario, where fully autonomous agents decide locally and sequentially their patrol routes according to the state of the system, as previously described. Beyond the coordination, which arises from the distributed communication of agents, it is also shown that the approach is robust to robot faults and communication failures.

Experiments were conducted in the floor 0 of the Institute of System and Robotics (ISR), in the University of Coimbra (UC), Portugal. Figure 16.2 shows the floor plan and the extracted topological map on top of the 67.85 × 26.15 m environment. The resulting topology is a noncomplete, connected and sparse graph, like most real world environments. In these tests, robots must overcome noisy sensor readings, localization issues, and even robot failures, which are usually ignored or not precisely modeled in simulation experiments. Therefore, a team of three Pioneer-3DX robots equipped with an Hokuyo URG-04LX-UG01 laser was used, as seen in Fig. 16.3.

The Pioneer 3-DX is a lightweight two-wheel differential drive robot for indoor use, equipped with two high-speed, high-torque, reversible-DC motors. Each motor has a high resolution optical quadrature shaft encoder for precise position, speed sensing, and advanced dead-reckoning. These robots are highly popular due to their versatility, reliability, and durability. They can operate continuously for 8–10 h, with a maximum load of 23 kg on top of the platform. In terms of dimensions, the robot has a diameter of 45.5 and 23.7 cms of height, as shown in Fig. 16.4. The robot easily handles small gaps and minor bumping, and its middle-size lends itself very well to navigation in tight quarters and cluttered spaces, such as classrooms,

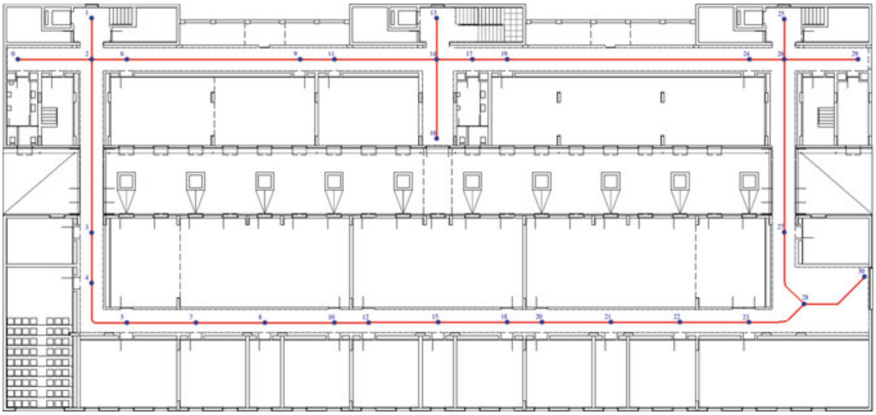

Fig. 16.2 Topological map of the “ISR-Floor0” Environment

Fig. 16.3 A team of Pioneer 3-DX robots

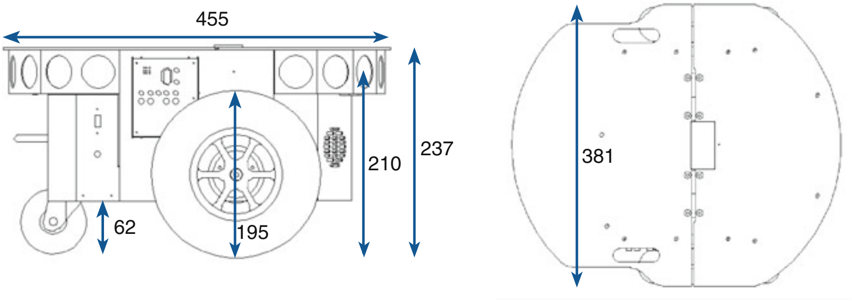

Fig. 16.4 Robot’s dimensions (in mm)

laboratories, and small offices. A laptop on top of each robot was used to extend its processing capability. Each laptop runs the ROS navigation stack with the Adaptive Monte Carlo (AMCL) algorithm for localization purposes, being responsible for controlling the robot’s motion. The ROS architecture running inside each robot is depicted in Fig. 16.5. A *patrolling* node, i.e., a ROS application, was added to the

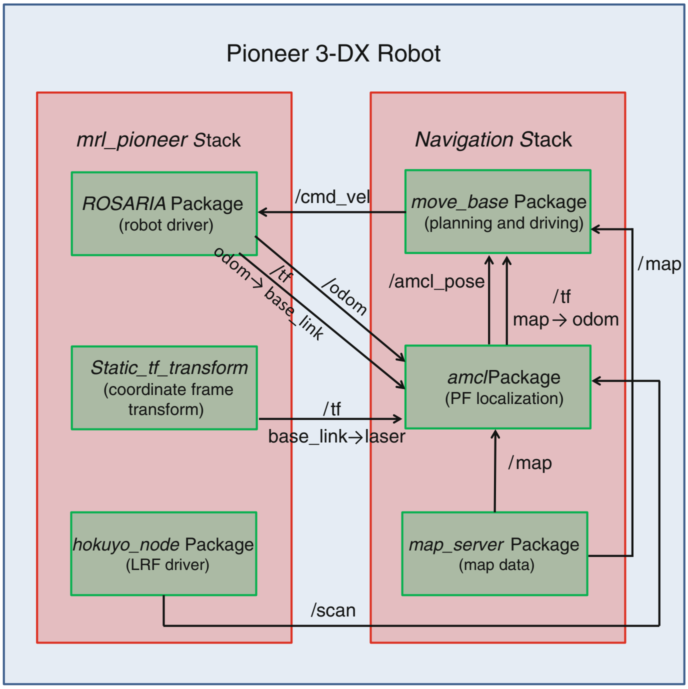

Fig. 16.5 ROS system running on each robot

architecture, having the responsibility to decide the robot's moves and send goals to the *move_base* node, which translates them into velocity commands for the robot base, in order to reach the given goal. All robots are limited to a maximum speed of 1 m/s. As for communication, a distributed publish/subscribe mechanism has been used, due to its built-in integration in ROS. Moreover, each robot runs its own ROS master node (*roscore*). Multimaster communication is provided using the *wifi_comm*[1] package. This means that there is no central point of failure in the system. Also, given that robots only share their current and future immediate goals, the bandwidth requirements are negligible even with large teams.

In the beginning of each test, the graph of the environment is loaded by every robot. A ROS node is responsible for advertising the start of the mission and collect results during the experiments. Not only is the average graph idleness along time $\overline{\mathcal{I}_{\mathcal{G}}}$ examined, but also the median $\widetilde{\mathcal{I}_{\mathcal{G}}}$, the standard deviation σ, and the maximum average idleness of a vertex along time, $\max(\overline{\mathcal{I}_{\mathcal{V}}})$.

Firstly, experiments with one, two, and three robots were conducted. Each experiment was repeated three times. Afterward, in order to further demonstrate the scalability of the approach, virtual robots were added to the team, and 3 trials with 6 agents $(3 + 3)$ and 9 agents $(3 + 6)$ were also conducted. It is noteworthy that

[1] Available at http://www.ros.org/wiki/wifi_comm.

adding virtual simulated agents to the physical teams of robots was only made possible by the hardware abstraction layer of ROS and its modular structure. Finally, to prove its robustness, experiments which included failures in the robots at different time instants are analyzed, as well as the impact of communication failures in the performance of the team. In all experiments, $|e_{\min}| = 7.5\,\text{m}$ has been used.

Aiming at comparing the total time of the mission (τ) in various conditions, each experiment finishes after four complete patrolling cycles. This stopping condition is adequate, as the $\overline{\mathcal{I}_\mathcal{G}}$ converges in all experiments. During the course of the experiments reported, the total estimated distance traveled by the robots was 23 km.

16.5.1 Scalability Experiments

Initially, experiments with one to three robots in the “ISR-Floor0” Environment were conducted. An overview of these results is shown in the top rows of Table 16.1. It can be seen that the $\overline{\mathcal{I}_\mathcal{G}}$ values, as well as the total mission time τ, decrease with team size, as expected. Additionally, the median is fairly close to the average idleness value, meaning that most data is spread around the mean.

An interesting result is the maximum average idleness, $\max(\overline{\mathcal{I}_\mathcal{V}})$, which is low for the case of one robot. This can be explained due to the existence of a main loop in the environment, which leads to fairly uniform visits to all vertices of the graph. In the cases of 2 and 3 robots, the value is higher compared to $\overline{\mathcal{I}_\mathcal{G}}$, because robots occasionally meet in the environment and need to coordinate by changing their heading direction. Consequently, no cycles are followed in such situations, and

Table 16.1 Overview of the scalability experiments

Team size	$\overline{\mathcal{I}_\mathcal{G}}$	$\max(\overline{\mathcal{I}_\mathcal{V}})$	$\overline{\mathcal{I}_\mathcal{G}}$	σ	τ
1	336.676	412.207	370.994	78.769	1648.828
	332.745	407.897	366.677	77.892	1631.590
	331.615	406.387	365.345	77.626	1625.550
2	168.921	309.455	137.267	64.210	1237.821
	180.761	296.085	180.293	56.064	1184.341
	170.267	328.300	146.890	62.603	1313.201
3	128.875	273.670	116.269	54.893	1094.682
	116.248	216.020	95.150	44.356	864.081
	112.954	200.030	101.923	36.066	800.121
6 (3 + 3)	71.097	152.625	65.483	27.130	610.500
	72.165	140.725	67.043	24.418	562.900
	77.332	150.145	72.938	27.350	600.580
9 (3 + 6)	48.623	102.305	47.395	16.499	409.220
	50.239	90.580	54.157	16.083	362.320
	51.687	105.12	52.271	19.622	420.480

All values in seconds

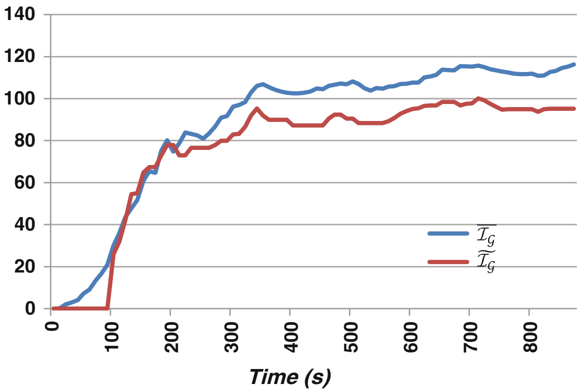

Fig. 16.6 Evolution of the idleness in a trial with 3 robots

the frequency of visits becomes less balanced. This can be confirmed by the standard deviation, which is around 23 % using 1 robot, and 35 and 37 % for a team size of 2 and 3 robots, respectively.

Figure 16.6 shows the evolution of the idleness in an experiment with 3 robots. After four patrolling cycles, $\overline{\mathcal{I}_\mathcal{G}}$ converges, meaning that it is no longer affected by the initial conditions, seeing as all vertices start with a null value of idleness.

The distributed patrolling method used supports an arbitrary high team size. However, we were limited by the physically available robots, which were $R = 3$. In order to test the approach with greater team size and further assess its scalability, virtual agents running in the Stage simulator were added to the team, resulting in a mixed team of real and simulated robots, which interact seamlessly via ROS. Three trials were conducted with a total of 6 agents comprising 3 physical robots and 3 simulated ones, and three additional trials were performed with a team size of 9, comprising 3 physical robots and 6 simulated ones. Similarly to the work in [23], the software layer is used unchanged both on real robots and in simulation.

Results in the bottom rows of Table 16.1 show that the overall values of $\overline{\mathcal{I}_\mathcal{G}}$, $\max(\overline{\mathcal{I}_\mathcal{V}})$, $\widetilde{\mathcal{I}_\mathcal{G}}$, σ and τ are within the expected, following the trend shown in the cases of two and three robots. In order to analyze how well the MRP strategy scales, Balch's speedup measure [31], a classical scalability metric, was calculated:

$$\upsilon(R) = \frac{\Psi(1)/R}{\Psi(R)}, \tag{16.24}$$

where $\Psi(R)$ is the performance for R robots, given by $\overline{\mathcal{I}_\mathcal{G}}$. Figure 16.7 presents a speedup chart using different team sizes. It can be seen that speedup and interference are negatively correlated, since the system enters progressively in sublinear performance ($\upsilon < 1$) with team size, due to the more frequent existence of spatial limitations, which in turn, increases the interference between robots, causing the performance to decrease. The interference between robots is measured as the overall

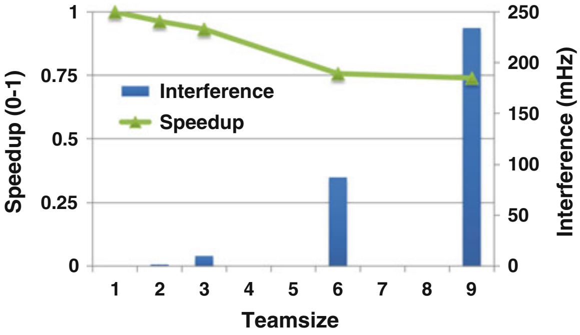

Fig. 16.7 Interference and Speedup against team size

frequency (in Hz) that different agents share nearby areas, having to avoid each other, in every experiment. These results show that the SEBS algorithm is able to scale to high number of robots, working independently of the team size. In addition, it is also illustrated that as the team size increases, the individual contribution of each robot decreases progressively as expected. This is in fact common to every MRP method tested in the literature.

16.5.2 Fault-Tolerance

One of the advantages of using autonomous robots with decision-making abilities is the absence of a centralized coordinator, which would represent a critical point of failure. A distributed robotic system, such as the one described enables redundancy, remaining functional if any of the agents fails.

In order to demonstrate the robustness of the patrolling method, three additional experiments using the Pioneer 3-DX robots were planned. In these experiments, a robot is shutdown during the course of the mission so as to understand the effect of the faults in the overall performance, as well as how the system evolves.

In the first experiment, a robot is shutdown after 200 s from the start of the mission. Similarly, in the second and third experiments, a robot is shutdown after 400 and 600 s, respectively. Teammates assume that the robot has failed when no message has been received from it for a period larger than 2 min.

Generally, Table 16.2 shows that the results obtained in the first experiment resembles those obtained with two robots, as most of the experiment is spent with only two agents due to the failure occurring near the beginning. On the other hand, the results of the second and third experiment are closer to those obtained with three robots, even though the performance is inferior, as expected.

Analyzing now the influence of the faults in the evolution of the results, one can verify that in all three cases when the shutdown occurs the values of $\overline{\mathcal{I}_{\mathcal{G}}}$ and $\widetilde{\mathcal{I}_{\mathcal{G}}}$ increase after a while, which is particularly visible in Fig. 16.8a, b.

Table 16.2 Experiments with 3 robots with failure of a robot in different instants of time (all values in secs)

Failure time (s)	$\overline{\mathcal{I}_\mathcal{G}}$	max($\overline{\mathcal{I}_\mathcal{V}}$)	$\overline{\mathcal{I}_\mathcal{G}}$	σ	τ
200	160.975	330.225	144.846	62.825	1320.901
400	140.128	232.290	134.177	45.934	929.161
600	135.209	235.700	139.797	41.262	942.801

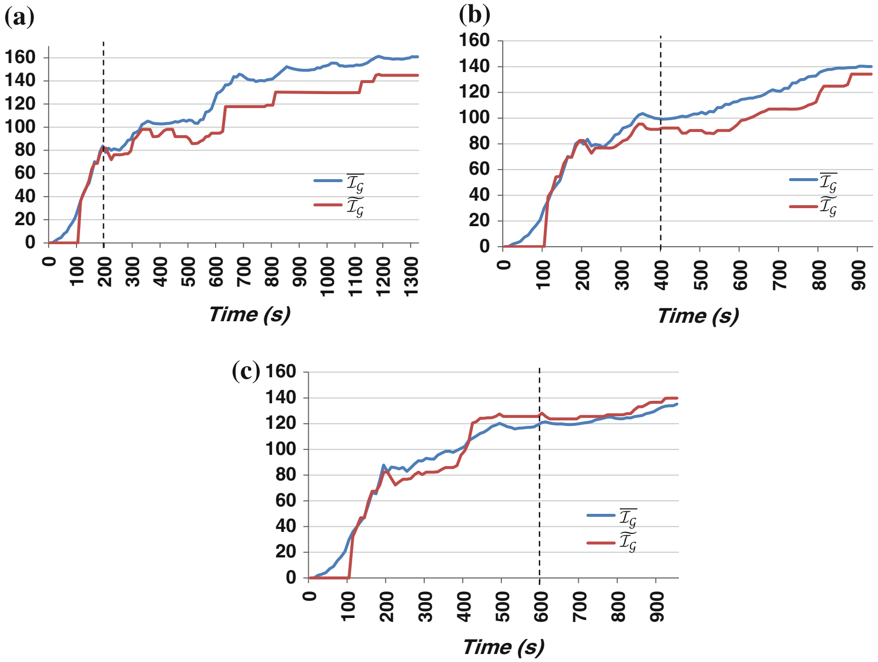

Fig. 16.8 Evolution of the idleness along time in experiments with robot failures **a** Failure at 200 s, **b** Failure at 400 s, **c** Failure at 600 s

Therefore, the MRP system using the proposed distributed strategy is resilient against robots' individual failures, presenting a graceful degradation of performance as long as at least one robot remains operational.

16.5.3 Influence of Communication Errors

The model proposed to solve the MRP problem assumes that agents are able to communicate seamlessly with other teammates during the course of the mission. However, this is not always the case, especially if a mobile *ad-hoc* network should be maintained and robots are occasionally far apart. In this section, multi-robot simulations in Stage were run in the "ISR-Floor0" map in order to test the robustness of the SEBS approach with different rates of communication failures.

When a message is not received by a robot, it does not update the instantaneous idleness time values and, consequently, it keeps incomplete information about the state of the system. This information becomes more incomplete with the increasing number of undelivered messages. Additionally, when robots are close to each other, if messages are not received, they may decide to move to the same places and interfere with their teammates' plans. The success of resolving such situations hugely depends on each robot's local planner and the ability to avoid dynamic obstacles. In these simulations, this is taken care by the ROS navigation stack.

In order to simulate different rates ξ of communication failures, a robot will ignore messages with a probability equivalent to ξ. In the reported experiments, the rates considered were: $\xi = \{0\,\%, 25\,\%, 50\,\%, 75\,\%, 100\,\%\}$. Furthermore, the system has also been tested allowing only local communication, restricted to robots within two edges of distance in the graph $\mathcal{G}$. This is a particular situation where it is ensured that robots are able to receive all the other nearby robots' intentions, and are thus able to coordinate themselves. Nevertheless, they are expected to make poor decisions as they are maintaining incomplete information about the system.

The chart in Fig. 16.9 presents an overview of the simulation results with different rates of communication failures, using team sizes of 2, 4, and 6 robots. Team performance is once again measured in terms of $\overline{\mathcal{I}_\mathcal{G}}$. The graph shows that performance gracefully degrades as ξ increases. The decrease of performance is approximately constant for the 25 %, 50 % and 75 % cases. However, when no communication is allowed, i.e., $\xi = 100\,\%$, the performance of the algorithm drops strongly, especially for larger teams, which are much more influenced by the lack of coordination in the multi-robot system, as robots constantly interfere with one another. This reduction of performance, especially for greater team sizes, is evident in the bars for $\xi = 100\,\%$; with 36.54 % for 2 robots, 51.30 % for 4 robots, and 66.84 % for 6 robots.

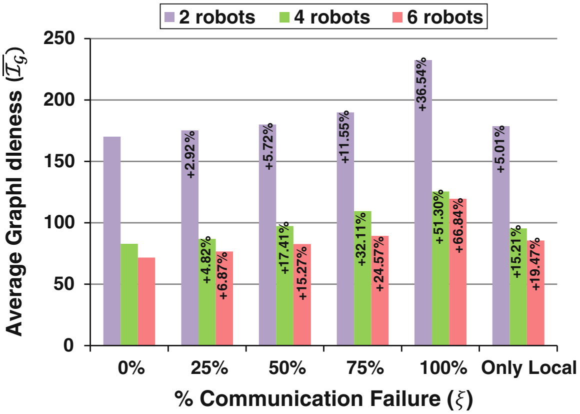

Fig. 16.9 Influence of communication failures in team performance

Also illustrated in the rightmost side of the chart is how performance is affected when communication is restricted to local interactions within 2 hops in $\mathcal{G}$. In this situation, robots are able to coordinate themselves by not competing to the same goals and not interfering with teammates. Despite that, they do not have contact with agents that are further away and, as a consequence, they will make uninformed decisions quite often. It can be seen that the system is able to perform well assuming such restrictions, especially for smaller team sizes. The performance obtained using only local communication closely resembles to that obtained when dropping 50 % of the messages for all team sizes.

In short, these results show that the approach is robust to communication failures and only slightly degrades its performance when communication error rate is moderate (e.g., 25 %). Evidently, the higher the rate of failures, the more affected performance is. Moreover, communication failures have more impact in the performance of systems with larger number of robots.

16.6 Conclusions and Future Work

The implementation of a distributed MRS for patrolling of an indoor infrastructure has been described. Breaking away from conventional techniques, our work goes beyond classical centralized approaches that rely on precomputed cyclic routes or partition schemes for MRP, giving the robots the autonomy to deal with uncertainty and select actions according to the state of the system at the time.

Previous results had shown the superior performance of the approach when compared with other state of the art MRP strategies in simulated environments [27]. This work confirms the potential and flexibility of employing Bayesian-based formalism to solve the MRP. The proposed approach accounts for the future immediate state of the system, preventing robots from competing to reach the same goals, consequently reducing interference and enhancing scalability.

Results have demonstrated that the approach is robust to robot faults and communication failures, and has the ability to adapt to constraints, e.g., different agent velocities, since the decision-making is done online with the information that each agent has collected about the system. Experiments were conducted using real robots and mixed teams of both virtual and real agents in a large indoor infrastructure proving the effectiveness of the approach and the potential to use it in the real world.

In the future, due to its flexibility, the model can be easily extended with more variables in order to employ it in different applications and/or use others sensors in the robots; e.g., readings from a temperature sensor may be included in the model, guiding robots towards heat sources in the environment. Moreover, even though the method allows different speed profiles, it would be interesting to study its influence in the mission, as well as the impact of unforeseen dynamic obstacles in the real world, such as people walking in the shared space and affecting the robot's navigation. Finally, we intend to devise an analytical method to compute the most adequate team size for a patrolling mission according to the environment topology and temporal constraints.

Acknowledgments This work has been supported by a Ph.D. grant (SFRH/BD/64426/2009), the CHOPIN research project (PTDC/EEA-CRO/119000/2010), and the Institute of Systems and Robotics (project Est-C/EEI/UI0048/2011), all of them funded by the Portuguese science agency "Fundação para a Ciência e a Tecnologia".

References

1. Webster's Online Dictionary (2014). Available at: http://www.webster-dictionary.org
2. Murphy R (2004) Human-robot interaction in rescue robotics. IEEE Trans Syst Man Cybern Part C Appl Rev 34(2):138–153
3. Rocha R P, Portugal D, Couceiro M, Araújo F, Menezes P and Lobo J (2013) The CHOPIN project: Cooperation between Human and robotic teams in catastrophic incident. In: Proceedings of the 2013 international symposium on safety, security and rescue robotics (SSRR 2013), Sweden, Linköping
4. Chevaleyre Y (2004) Theoretical analysis of the multi-agent patrolling problem. In : Proceedings of the 2004 international conference on agent intelligent technologies (IAT'04), China, Beijing, pp 30–308
5. Smith S, Rus D (2010) Multi-robot monitoring in dynamic environments with guaranteed currency of observations. In: Proceedings of the 49th IEEE conference on decision and control, USA, Atlanta, Georgia, pp 514–521
6. Pasqualetti F, Franchi A, Bullo F (2012) On cooperative patrolling: optimal trajectories, complexity analysis, and approximation algorithms. IEEE Trans Rob 28(3):592–606
7. Fazli P, Davoodi A and Mackworth AK (2013) Multi-robot repeated area coverage. In: Autonomous robots, 34(4), Springer Science, May 2013, pp 251–276
8. Portugal D, Rocha R (2010) MSP Algorithm: multi-robot patrolling based on territory allocation using balanced graph partitioning. In: Proceedings of 25th ACM symposium on applied computing (SAC 2010), Special track on intelligent robotic systems, Switzerland, Sierre, pp 1271–1276
9. Pippin C, Christensen H and Weiss L (2013) Performance based task assignment in multi-robot patrolling. In Proceedings of the 2013 ACM symposium on applied computing (SAC'13), Portugal, Coimbra, pp 70–76
10. Machado A, Ramalho G, Zucker J and Drogoul A (2003) Multi-agent patrolling: an empirical analysis of alternative architectures. In: Multi-agent-based simulation II, Lecture notes in computer science, vol 2581. Springer, pp 155–170
11. Sempé F and Drogoul A (2003) Adaptive patrol for a group of robots. In: Proceedings of the 2003 IEEE/RSJ international conference on intelligent robots and systems (IROS'2003), Las Vegas, Nevada, USA
12. Sampaio P, Ramalho G and Tedesco P (2010) The gravitational strategy for the timed patrolling. In: Proceedings of the IEEE international conference on tools with artificial intelligence (ICTAI'10), Arras, France, pp 113–120
13. Marino A, Parker L, Antonelli G, Caccavale F (2009) Behavioral control for multi-robot perimeter patrol: a finite state automata approach. In: Proceedings of the IEEE international conference on robotics and automation (ICRA'09), Kobe, Japan, pp 831–836
14. Aguirre O, Taboada H (2012) An evolutionary game theory approach for intelligent patrolling. In: Procedia computer science, vol 12, part II, Elsevier, pp 140–145
15. Yanovski V, Wagner IA, Bruckstein AM (2003) A distributed ant algorithm for efficiently patrolling a network. Algorithmica 37:165–186

16. Marier J, Besse C and Chaib-draa B (2010) Solving the continuous time multiagent patrol problem. In: Proceedings of the 2010 IEEE international conference on robotics and automation (ICRA'10), Anchorage, Alaska, USA
17. Keskin BB, Li S, Steil D, Spiller S (2012) Analysis of an integrated maximum covering and patrol routing problem. In: Transportation research Part E: Logistics and transportation, vol 48, Elsevier, pp 215–232
18. Portugal D, Rocha R (2011) A survey on Multi-robot patrolling algorithms. Technological innovation for sustainability, IFIP advances in information and communication technology series, vol 349. Springer, Berlin, pp 139–146
19. Sak T, Wainer J and Goldenstein S (2008) Probabilistic multiagent patrolling. In: Advances in artificial intelligence, proceedings of the 19th Brazilian symposium on artificial intelligence (SBIA 2008), Lecture Notes in Computer Science, vol 5249, Springer, pp 124–133
20. Agmon N, Kaminka G, Kraus S (2011) Multi-robot adversarial patrolling: facing a full-knowledge opponent. J Artif Intell Res (JAIR) 42:887–916
21. Basilico N, Gatti N, Rossi T, Ceppi S and Amigoni F (2009) Extending algorithms for mobile robot patrolling in the presence of adversaries to more realistic settings. In: Proceedings of the 2009 IEEE/WIC/ACM international conference on intelligent agent technology (IAT'09), Milan, Italy, pp 557–564
22. Pita J, Tambe M, Kiekintveld C, Cullen S, Steigerwald E (2011) GUARDS-innovative application of game theory for national airport security. In: Proceedings of the 22nd international joint conference on artificial intelligence (IJCAI'11), vol 3. Spain, Barcelona, pp 2710–2715
23. Iocchi L, Marchetti L and Nardi D (2011) Multi-robot patrolling with coordinated behaviours in realistic environments. In: Proceedings of the international conference on intelligent robots and systems (IROS'2011), San Francisco, CA, USA, pp 2796–2801
24. Portugal D and Rocha RP (2013) Retrieving topological information for mobile robots provided with grid map. In: Agents and artificial intelligence, communications in computer and information science (CCIS) series, vol 358. Springer, Berlin, pp 204–217
25. Portugal D, Rocha RP (2011) On the performance and scalability of multi-robot patrolling algorithms. In: Proceedings of the 2011 IEEE international symposium on safety, security, and rescue robotics (SSRR 2011), Japan, Kyoto, pp 50–55
26. Portugal D and Rocha RP (2013) Multi-robot patrolling algorithms: examining performance and scalability. In: Advanced robotics journal, special issue on safety, security, and rescue robotics, 27(5), pp 325–336
27. Portugal D and Rocha RP (2012) Decision methods for distributed multi-robot patrol. In: Proceedings of the 10th international symposium on safety, security and rescue robotics (SSRR'2012), College Station, Texas, USA
28. Vaughan R (2008) Massively multi-robot simulation in stage. J Swarm Intell 2(2–4):189–208
29. Quigley M, Gerkey B, Conley K, Faust J, Foote T, Leibs J, Berger E, Wheeler R, Ng A (2009) ROS: an open-source robot operating system. In: Proceedings of the IEEE international conference on robotics and automation (ICRA'2009), workshop on open source software, Japan, Kobe
30. Jansen F and Nielsen T (2007) Bayesian networks and decision graphs, 2nd edition, Springer
31. Balch T, Arkin R (1994) Communication in reactive multiagent robotic systems. Auton Robots 1(1):27–52