2014 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS 2014)
September 14-18, 2014, Chicago, IL, USA

Finding Optimal Routes for Multi-Robot Patrolling in Generic Graphs

David Portugal[1], Charles Pippin[2], Rui P. Rocha[1] and Henrik Christensen[3]

Abstract— **Multi-robot patrolling is a problem that has important applications in security and surveillance. However, the optimal task assignment is known to be NP-hard. We consider evenly spacing the robots in a cyclic Traveling Salesman Problem (TSP) tour or partitioning the graph of the environment. The trade-off in performance, overall team travel cost and coordination is analyzed in this paper. We provide both a theoretical analysis and simulation results across multiple environments. The results demonstrate that generally cyclic-based strategies are superior, especially when small teams are used but at the expense of greater team cost, whereas partitioning strategies are especially suitable for larger teams and unbalanced graph topologies. The reported results show that graph topology and team size are fundamental to determine the best choice for a patrol strategy.**

I. INTRODUCTION

Advances on autonomous mobile robots have been evident in the last couple of decades. In particular, the patrolling problem with a team of cooperative agents has received much focus. The problem, which is also known as *repetitive sweeping*, has unquestionable utility in society and finds its applications in surveillance systems, infrastructure security and inspection, search and rescue, mine clearing, military operations, environmental monitoring, intelligent transportation, household cleaning, and several other areas. Being monotonous, these tasks may also be dangerous. Thus, improving safety and reducing fatigue is a major advantage of multi-robot patrolling systems.

In this context, robots are required to continuously travel in the environment, and the key challenge is to design efficient routes in order to optimize a certain performance criterion. Like most existing work in the literature [1–4], it is assumed that robots are homogeneous, travel with constant speed, and are expected to visit every strategic position of the environment. Therefore, having adequate sensing range, complete coverage of the environment is achieved by visiting all the important locations in the area.

Despite several recent contributions to the problem, one question still remains open: what is the optimal patrol strategy for a given generic environment using R robots? Using the worst idleness criterion (*cf.* Section III), it has been shown that the problem is NP-Hard. In this work, four strategies based on the concepts of tours and graph partitioning are analyzed. This allows us to understand which approach best suits a given generic environment with an arbitrarily high number of robots. We address the theoretical results known so far and pose problems that are still open in this area of research. In addition, we extract results and examine graphs with different connectivity and different team size in order to draw conclusions and discuss implementation issues on real world robots.

II. RELATED WORK

Several strategies for teams of robots in patrolling missions have been proposed in the last decade. Pioneering work was done in [5], where many intuitive techniques using agents that were guided towards places left unvisited for a long time were described and empirically compared in simulations. Graph-based coverage was considered and the notion of idleness was introduced for the first time, being since then widely used in the literature, along with visit frequency of important locations [6].

A wide diversity of concepts have been applied to multi-robot area patrol; *e.g.*, Task Allocation [7], Gaussian Processes Theory [8]; and lately, an effort for real-world validation of these systems has been evident [4]. Additionally, different coordination methods for the team of agents have been studied, such as centralized deterministic [10] and distributed probabilistic methods [9]. While the former methods have the advantage of reaching near-optimal solutions, often with performance guarantees due to the global knowledge assumed, the latter provides interesting features derived from the robots' autonomy, like the potential to adapt to different situations arising during the mission, being more robust to failures and less predictable by an external observer. In this article, we will focus on centralized and deterministic coordination methods, where the routes of the robots are computed prior to the mission, since we are concerned in obtaining optimal patrolling performance.

To that end, important theoretical contributions on the problem have been presented by [1] and [2]. Considering the worst idleness criterion (*cf.* Section III), Pasqualetti et al. showed that the multi-robot patrolling problem is NP-Hard, and Chevaleyre proved that it can be optimally solved with a single robot by finding a TSP tour in the graph that describes the environment to patrol. As for the multi-robot case, it was shown that partition-based strategies may perform better than a TSP cyclic strategy "for graphs containing long edges" [1]. Furthermore, Pasqualetti et al. addressed

This work was supported by a PhD scholarship (SFRH/BD/64426/2009), the CHOPIN research project (PTDC/EEA-CRO/119000/2010) and by the 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). This work was also supported internally by the Georgia Tech Research Institute.

[1]D. Portugal and R.P. Rocha are with the Institute of Systems and Robotics, Dept. Electrical and Computers Eng., University of Coimbra, 3030-290 Coimbra {davidbsp,rprocha} at isr.uc.pt

[2]C. Pippin is with the Georgia Tech Research Institute, Georgia Institute of Technology, Atlanta, GA, USA {pippin} at gatech.edu

[3]H. Christensen is with the College of Computing, Georgia Institute of Technology, Atlanta, GA, USA {hic} at cc.gatech.edu

978-1-4799-6934-0/14/$31.00 ©2014 IEEE

approximation algorithms to obtain known bounds related to the optimal result and focused on specific graph instances. Unlike [2], in this work we address generic graphs describing real world environments and propose heuristics to obtain near-optimal results. These heuristics consistently result in superior trajectories (or at least equally good) to constant factor approximation algorithms in all graphs tested.

Therefore, the main goal of this work is to further understand how two classical types of strategies – graph partition and cyclic-based techniques – perform in generic graphs, by making use of efficient graph theory algorithms and analyzing comparative results. We prove that graph connectivity is not the only parameter that affects the suitable choice of a strategy, and team size should also be considered. Moreover, we discuss implementation issues on real world robots and provide a testing tool to the community to extend the results presented herein, in order to infer about the optimality on any generic graph.

III. Preliminaries and Problem Formulation

As referred before, it is common to represent the area to patrol by a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $v_i \in \mathcal{V}$ vertices and $e_{i,j} \in \mathcal{E}$ edges. Therefore, $\mathcal{G}$ corresponds to the topological map for the patrolling mission, which is obtained from a metric representation, assumed to be known *a priori*, by means of a graph extraction algorithm [9]. In this representation, vertices correspond to important places or landmarks and edges represent the connectivity between those locations. The cost of an edge $|e_{i,j}|$ is defined by the distance between vertex v_i and v_j. A path x is composed of an array of vertices in $\mathcal{V}$ and the cost of a path is given by $L(x)$.

Seeing as the topological maps considered in this article represent real world 2D environments, it is assumed that $\mathcal{G}$ has the following properties:

- Undirected, where $|e_{i,j}| = |e_{j,i}|$, and the edge weights satisfy the triangle inequality;
- Connected, where $\forall v_h, v_i \in \mathcal{V}, \exists x = \{v_h, ..., v_i\}$;
- Simple, where two neighbor vertices v_i and v_j are connected by a unique edge $e_{i,j}$ and no graph loops exist;
- Planar, where a pair of edges $e_{g,h}, e_{i,j} \in \mathcal{E}$ never crosses each other.

As a consequence of these properties, $\mathcal{G}$ is usually non-complete, *i.e.* for every pair v_h, v_i of $\mathcal{V}$ there may not exist an edge $|e_{h,i}|$ connecting each pair of vertices. Additionally, we address any generic planar graph, as opposed to specific instances such as chain graphs, cyclic or acyclic graphs, tree graphs, Hamiltonian graphs, *etc.*

The multi-robot patrolling problem is then reduced to find R trajectories $\Pi = \{\pi_1, ..., \pi_R\}$ for each robot in order to visit frequently all vertices $v_i \in \mathcal{V}$ with respect to a predefined optimization criterion. Thus, the idleness of a vertex $v_i \in \mathcal{V}$ in time step t is defined as:

$$\mathcal{I}_{v_i} = t - t\iota_i, \tag{1}$$

where $t\iota_i$ corresponds to the last time instant when the vertex v_i was visited by any robot of the team. Consequently, the worst idleness $\mathcal{WI}$ corresponds to the largest idleness value for all $v_i \in \mathcal{V}$ that occurred during the patrolling task that lasted τ time units:

$$\mathcal{WI} = \max_{v_i \in \mathcal{V}} \max_{t \in [0,\tau]} \mathcal{I}_{v_i}. \tag{2}$$

Using $\mathcal{WI}$ as the optimization criterion, the multi-robot patrolling problem is formulated as follows:

Problem 1 (Multi-Robot Patrol). *Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a team of R robots, find a policy Π^* such that:*

$$\Pi^* = \arg\min_{\Pi}(\mathcal{WI}_\Pi), \tag{3}$$

wherein $\mathcal{WI}_\Pi$ is the Worst Idleness when using policy Π.

Remark 1 (Computational Complexity). *It has been shown that Problem 1 is NP-Hard by reduction from the Traveling Salesman Problem (cf. Theorem II.1 in [2]).*

Two classical types of strategies have been used previously to obtain optimal or near-optimal results: cyclic-based and partition-based strategies [1], [2]; and considering the worst idleness criterion, superior approaches are still not known. Using a variety of graph theory concepts, it is possible to devise patrolling trajectories for the team of robots based on these two classes of approaches, which are defined below.

Definition 1 (Cyclic-based Strategy). *Given a closed walk $\pi_{Cyc} = \{v_a, v_b, ..., v_a\}$ in $\mathcal{G}$, such that $\forall v_i \in \mathcal{V} : v_i \in \pi_{Cyc}$ and possibly visiting vertices more than once, a Cyclic-based strategy Π_{Cyc} places R agents, that move at the same speed, equally spaced along π_{Cyc}, while keeping a constant gap between them.*

Definition 2 (Partition-based Strategy). *Being $P_r \in \mathcal{V}$ a partition of the environment assigned to robot r and given a set of disjoint partitions $P = \{P_1, ..., P_R\}$ in $\mathcal{G}$, such that $\cup_{r=1}^{R} P_r = \mathcal{V}$ and $P_i \cap P_j = \emptyset$ with $i \neq j$, in a Partition-based strategy Π_P, each agent r visits the vertices of a single partition P_r, by following a strategy π_r.*

In the next two sections, we present two cyclic-based and two partition-based approaches. In cyclic strategies, it is necessary to compute a patrolling-effective closed walk π_{Cyc}. As for partitioning strategies, we focus not only on computing an effective set of partitions P, but also on defining each agent's strategy π_r on each partition P_r.

For both cyclic and partition-based approaches, we test a well-known method in the literature with a constant factor approximation and propose one heuristic method as an alternative. These heuristics are employed to obtain closer results to the optimum and to further understand the potential of each class of strategy in graphs with different connectivity and using different team sizes.

IV. Cyclic-based Strategies

In a cyclic-based strategy, the time taken for an agent to visit a vertex for the second time is at most $L(\pi_{Cyc})/c$, where c is the average agent's speed and $L(\pi_{Cyc})$ is the length of the walk π_{Cyc}. Without lack of generality, let us assume

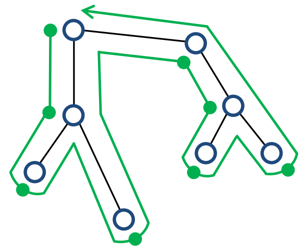


Fig. 1. Obtaining a closed walk using a MST (Algorithm 1).

that agents move at unitary speed. The worst idleness of any cyclic based-strategy ($\mathcal{WI}_{\Pi_{Cyc}}$, or simply $\mathcal{WI}_{Cyc}$), using a single agent, is given by:

$$\mathcal{WI}_{Cyc} = L(\pi_{Cyc}), \quad R = 1, \tag{4}$$

since agents are equally spaced along π_{Cyc}, we can extend (4) for a multi-robot situation with R robots:

$$\mathcal{WI}_{Cyc} = \frac{L(\pi_{Cyc})}{R}. \tag{5}$$

Clearly, by minimizing $L(\pi_{Cyc})$, *i.e.*, finding the smallest π_{Cyc} walk that visits every vertex of $\mathcal{G}$, $\mathcal{WI}$ becomes minimal in (5). Consequently, it becomes evident that the Traveling Salesman Problem (TSP) solution for $\mathcal{G}$ is the best possible solution among all cyclic-based strategies:

$$\Pi^*_{Cyc} = \Pi_{TSP}. \tag{6}$$

TSP is a classical NP-complete problem, and no polynomial time algorithm is known to compute an optimal solution to it. In this section, two different methods for approximating a metric TSP tour[1] in a generic graph $\mathcal{G}$ are discussed.

The first method is a well-known approximation for the metric TSP tour, based on the Minimum Spanning Tree (MST) concept, as shown in Fig 1. Consider the following algorithm.

Algorithm 1 (MST Tour Approximation)**.**

i) *Find a Minimum Spanning Tree $\mathcal{T}$ in $\mathcal{G}$.*

ii) *Conduct a depth-first search (DFS) to visit all $v_i \in \mathcal{T}$ in a depth-first order.*

iii) *Build a closed walk π_{MSTt} that visits all vertices, following the order of DFS discovery.*

iv) *Equally space R moving robots along π_{MSTt}.*

Theorem 1 (Constant factor approximation)**.** *MST Tour is a 2-approximation for the metric TSP.*

Proof. Let $L(\pi_{TSP})$ be the cost of an optimal TSP tour. Recall that by removing an edge from π_{TSP}, one obtains a spanning tree. Therefore, the Minimum Spanning Tree provides a lower bound for the optimal tour: $L(\mathcal{T}) \leq L(\pi_{TSP})$. Notice that the length of a depth-first tour of the connected tree $\mathcal{T}$ equals twice the sum of the length of the edges of $\mathcal{T}$: $L(\pi_{MSTt}) = 2L(\mathcal{T})$. Hence, $L(\pi_{MSTt}) \leq 2L(\pi_{TSP})$. ■

[1]In the metric TSP, the edge costs satisfy the triangle inequality, *i.e*, for any $v_h, v_i, v_j \in \mathcal{V}, e_{h,j} \leq e_{h,i} + e_{i,j}$.

Corollary 1. *It immediately follows that the worst idleness of the MST tour $\mathcal{WI}_{\Pi_{MSTt}}$ with R agents is at most 2 times the worst idleness of the optimal TSP tour $\mathcal{WI}_{\Pi_{TSP}}$ with R agents.*

Algorithm 1 can quickly obtain a 2-approximate solution for the TSP in feasible time. In fact, our implementation (*cf.* section VI), uses Kruskal's algorithm in step i to compute the MST [12], which runs in $O(|\mathcal{E}| \log |\mathcal{V}|)$ time. Despite the performance guarantees given by $MSTt$, Algorithm 1 does not lead in general to an optimal TSP tour. This is clear in the results reported in the next section. For a generic graph $\mathcal{G}$, we have tested an additional cyclic-based algorithm.

Algorithm 2 (Heuristic to approximate the TSP Tour)**.**

i) *Create a complete unique graph $\mathcal{G}_C = (\mathcal{V}_C, \mathcal{E}_C)$, by copying all vertices and edges of $\mathcal{G}$, and for all non-existing edges between pairs of vertices v_h, v_i in $\mathcal{G}$, create a unique edge $e_{h,i}$ in $\mathcal{G}_C$, where $|e_{h,i}| = \text{dijkstra}(v_h, v_i)$.*

ii) *Compute an efficient heuristic for the TSP in $\mathcal{G}_C$.*

iii) *Convert the TSP tour obtained in $\mathcal{G}_C$ into a minimum cost closed walk $\pi_{\mathcal{H}TSP}$ in $\mathcal{G}$ by translating each edge $|e_{h,i}| \in [\mathcal{E}_C \cap \overline{\mathcal{E}}]$ of the TSP tour into a shortest path of $\mathcal{G}$ in $\pi_{\mathcal{H}TSP}$.*

iv) *Equally space R moving robots along $\pi_{\mathcal{H}TSP}$.*

Before proceeding, it is important to demonstrate that solving the TSP in a complete graph $\mathcal{G}_C$ is equivalent to solving the minimum cost closed walk problem in a non-complete graph $\mathcal{G}$, where ocasionally repeating visits to vertices is allowed.

Theorem 2 (Minimum Cost Closed Walk)**.** *The TSP tour of the complete graph $\mathcal{G}_C$ is equivalent to the minimum cost closed walk in $\mathcal{G}$.*

Proof. When decoding a TSP tour of $\mathcal{G}_C$ into a closed walk $\pi_{\mathcal{H}TSP}$ in $\mathcal{G}$ (step *iii* of Alg. 2), clearly all edge costs are preserved. Assume now that there is a shorter closed walk π_s such that $L(\pi_s) < L(\pi_{\mathcal{H}TSP})$ in $\mathcal{G}$. Translate π_s into a tour in $\mathcal{G}_C$ by selecting the vertices in the order in which they appear first. This would imply a tour in $\mathcal{G}_C$ shorter than the optimal TSP tour. Hence, we have a contradiction. ■

Algorithm 2 presents an heuristic solution $\Pi_{\mathcal{H}TSP}$ for the minimum cost closed walk with R agents. It should be noted that we use Dijkstra's algorithm to obtain the shortest path in $\mathcal{G}$ and, in our implementation (see section VI), the heuristic chosen to approximate the TSP in a complete graph (step *ii*) is a genetic algorithm named TSP_GA[2]. Despite not having performance guarantees, the results have shown that the approach is able to quickly compute the optimal solution for every graph tested in this work. The graphs considered have several tens of vertices, similarly to those that commonly represent real world buildings.

[2]TSP_GA has been developed by Joseph Kirk. For more information, it is openly available at: `http://www.mathworks.com/matlabcentral/fileexchange/13680`

V. Partition-based Strategies

In the past, it has been shown [1–4] that partitioning strategies may have superior performance than cyclic ones. In fact, we recall two important contributions by Chevaleyre [1]:

i) The optimal partition-based strategy Π_P^* is a disjoint partition, where each agent behaves optimally inside each subgraph, by running a TSP tour on it (*i.e.*, a minimum cost closed walk).

ii) Cyclic strategies are not suited for graphs containing long edges, as shown by the following result: $\mathcal{WI}_{Cyc^*} \leq \mathcal{WI}_{P^*} + 3 \cdot \max |e_{i,j}|$.

These two contributions are of high importance in the multi-robot patrolling literature. Nevertheless, they lead to two important follow-up questions. In *i)*, each agent's strategy π_r becomes evident, however how should one optimally compute a set of R partitions P in the first place? Moreover, in *ii)*, the inequality can be verified with $\mathcal{WI}_{Cyc^*} > \mathcal{WI}_{P^*}$ or $\mathcal{WI}_{Cyc^*} < \mathcal{WI}_{P^*}$. So, which strategy should one choose for a given graph $\mathcal{G}$ patrolled by R agents? In this section we present two partition-based strategies to address the first question. Later, we address the second question (section VI).

In a partition-based strategy, each agent follows a minimum cost closed walk π_r in the subgraph induced by partition P_r in $\mathcal{G}$. Thereby, $\mathcal{WI}$ on each partition is given by (4): $\mathcal{WI}_{\pi_r} = L(\pi_r)$. Note that all partitions are disjoint, hence each vertex is always visited by the same robot. Since partitions are patrolled in parallel, $\mathcal{WI}$ considering unitary agent's speed without lack of generality, is given by the maximum length of any tour π_r:

$$\mathcal{WI}_P = \max_{r \in R} L(\pi_r). \tag{7}$$

Clearly, by minimizing the partition tour π_r with maximal cost, we obtain the best possible solution among all partition-based strategies Π_P^*. However, classical graph-partitioning is a NP-hard problem, being usually studied in the context of parallel computing and clustering applications. For high performance in such systems, regions should be identically sized, *i.e.*, each partition should have the same number of vertices $|P_1| \simeq ... \simeq |P_R|$; and the link between regions should be small, *i.e.*, the edges that connect different partitions should have minimal cost. While this may yield a satisfactory solution for our problem, it is necessary to consider two fundamental differences to such applications.

Firstly, due to (7), instead of identically sized partitions, we aim at obtaining trajectories with balanced cost $L(P_1) \simeq ... \simeq L(P_R)$ so as to minimize the partition tour π_r with maximal cost and consequently, $\mathcal{WI}_P$. Secondly, since the edges between partitions are not traversed by any robot, the cut should ideally be conducted on long edges. Hence, the following problem is defined.

Problem 2 (Min-Max Cost Closed Walk)**.** *Given a generic graph $\mathcal{G}$, find a set of disjoint partitions $P = \{P_1, ..., P_R\}$ in $\mathcal{G}$ such that:*

$$P^* = \arg\min_P (\max L(\pi_r)). \tag{8}$$

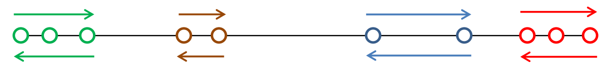


Fig. 2. Four optimal closed walks on a chain graph.

Theorem 3 (Computational Complexity)**.** *The min-max cost closed walk problem is NP-hard.*

Proof. When $R = 1$, this problem is equivalent to finding the minimal cost closed walk, which in its turn, is equivalent to the Traveling Salesman Problem (see Theorem 2). Since the TSP is a NP-hard problem, by restriction the min-max cost closed walk problem is also NP-hard. ∎

Similarly as before, in order to solve Problem 2, we test one algorithm with known performance ratio and an evolutionary heuristic technique. For the first partition-based algorithm, we will apply a previously known result. The authors in [2] have proposed an optimal min-max cost closed walk partition for the particular case of a chain graph, *e.g.*, Fig 2, which was called an "Optimal Left-Induced partition". By extending this result to generic graphs, the following approximation method has been proposed.

Algorithm 3 (Left-Induced Partition on Generic Graphs)**.**

i) *Find an open walk, with at most $2|\mathcal{V}| - 4$ edges, that visits all $v_i \in \mathcal{V}$ of $\mathcal{G}$.*

ii) *Construct a chain graph Γ, equivalent to the open walk in i).*

iii) *Compute the optimal left-induced partition with R agents in Γ.*

iv) *In order to create a solution Π_{LIP}, assign a partition P_r to each of the R agents and have them patrolling each region back and forth.*

Remark 2 (Performance Ratio)**.** *It has been shown that Algorithm 3 leads to the following performance guarantee with respect to the optimal solution [2]:*

$$\mathcal{WI}_{LIP} \leq 8 \left(\frac{|V| - 2}{|V|} \right) \eta \, \mathcal{WI}_{\Pi^*}, \tag{9}$$

with:

$$\eta = \frac{\max |e_{i,j}|}{\min |e_{i,j}|}.$$

Note that an open walk in $\mathcal{G}$ (step *i*) can be obtained from a MST, by starting from any leaf of the tree and stopping when all vertices have been visited. Nonetheless, in our implementation we have considered instead the result given by Algorithm 2, which corresponds to a TSP tour on a complete graph $\mathcal{G}_C$. Clearly, an open walk with equal or inferior cost can be obtained by removing the longest edge of the TSP tour in $\mathcal{G}_C$ and translating it into an open walk in $\mathcal{G}$. Moreover, with at most $2|\mathcal{V}| - 4$ edges, the performance bounds indicated by Remark 2 remains.

Despite the performance guarantees of Algorithm 3, given the high dependence on η, the previous algorithm is rarely expected to reach an optimal solution. Thus, an additional partition-based evolutionary algorithm is proposed to solve the multi-robot patrol problem.

Algorithm 4 (Evolutionary Heuristic to approximate the Min-Max Cost Closed Walk Problem)**.**

i) *Compute a set of R initial partitions using a classical multi-way graph-partitioning approach on $\mathcal{G}$ and start a counter* iter=0*.*

ii) *Initially set* global_best = $L(P_n)$ *as the length of the partition with maximal cost* $P_n = \arg\max L(P_{init})$.

iii) *Swap a vertex from P_n with a random neighbor partition P_m.*

iv) *Make sure P_n stays connected. Otherwise, randomly keep one of the disjoint parts and swap all the others to P_m.*

v) *Assign $P_n = P_m$, if $|P_m| > 1$ and P_m has not been used before. Otherwise choose randomly for P_n a partition that has not been chosen as P_n before.*

vi) *If* $\max L(P_i) > \Phi \max L(P_{init})$ *or all partitions have already been used as P_n, reset* $P_n = \arg\max L(P_{init})$.

vii) *Save solution* global_best = $\max L(P_i)$ *if* $\max L(P_i) <$ global_best*. Increment* iter*.*

viii) *Repeat steps iii - vii, while* iter $\leq$ MaxSteps*.*

xix) *Build $\Pi_{E\mathcal{H}P}$ by considering the set of partitions that generated* global_best *and use Algorithm 2 to compute a min cost closed walk for each agent r.*

In Algorithm 4, vertex swaps in step *iii)* correspond to a mutation mechanism on the current solution. Additionally, the concept of "survival of the fittest" is applied, as the best solution found is kept during run time. In step *i)*, we make use of METIS multi-way partitioning [13]. Furthermore, it is necessary to dimension the exploration factor Φ. A low value ($\Phi < 2.5$) may not let the approach explore the search space conveniently, possibly falling into local minima. High values ($\Phi > 5$) may lead the approach to spend too much time generating solutions with low quality. In our experiments, we have used $2.5 \leq \Phi \leq 5.0$ and MaxSteps ≤ 15000.

In the next section, we present a discussion of results using this partition-based strategy and the three other previously presented strategies in graphs with different connectivity and teams of patrolling agents with different sizes.

VI. Results and Discussion

In this section, the three topological maps $\mathcal{G}_a$, $\mathcal{G}_b$ and $\mathcal{G}_c$ in Fig. 3 are employed in order to test all previously described strategies for multi-robot patrol. These present different algebraic connectivity or Fiedler value [14], a well-known metric of the connectivity of a graph. These topologies are classified as: lowly (A), mildly (B) and highly (C) connected, having a Fiedler value of $\lambda_a = 0.0080$, $\lambda_b = 0.0317$ and $\lambda_c = 0.1313$, respectively. Note, for example, that $\mathcal{G}_a$ has several dead-ends, *i.e.*, vertices with degree one[3]. On the other hand $\mathcal{G}_c$ is the most connected of the three, with a maximum degree of 4. Despite being a highly connected graph in the context of a patrolling mission, $\mathcal{G}_c$ is far from being complete (each vertex would need to have degree 24) and may eventually be considered a sparse graph in other applications.

Even though cyclic-based and partition-based strategies are expected to perform differently with the connectivity of $\mathcal{G}$, in this paper we also aim to analyze performance among teams with different sizes. To this end, all approaches have been tested with $R \in [1, 20]$. Our implementation consists of building the routes for the MST Tour approximation (Π_{MSTt}), the heuristic for the TSP tour ($\Pi_{\mathcal{H}TSP}$), the Left Induced partition-based strategy (Π_{LIP}) and the evolutionary partition-based heuristic ($\Pi_{\mathcal{EH}P}$), and compute the worst idleness $\mathcal{WI}$ for an arbitrary R, using (5) for cyclic-based strategies and (7) for partition-based strategies. Furthermore, all methods used in the paper are made available to let the reader test other graph instances as desired[4].

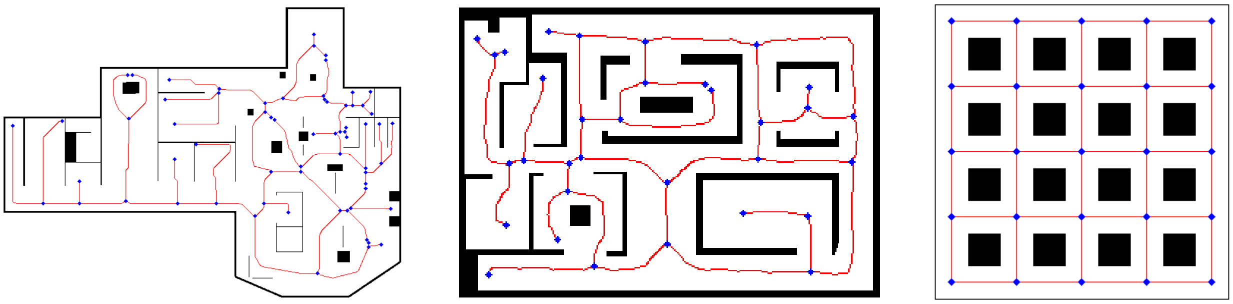
Table I presents the overall results. Since one of the main goals of this work is to understand which class of strategy is more suited given a generic graph $\mathcal{G}$ and team size R, the best results over 25 trials for the partition-based strategies were saved. This is because Π_{LIP} depends on the choice of an open walk, which may differ in each trial and $\mathcal{EH}P$ being an evolutionary algorithm may not always reach an optimal solution. As for cyclic-based strategies, this was not necessary because $MSTt$ always returns a spanning tree tour with minimal cost and $\mathcal{H}TSP$, as seen before, easily computes one optimal minimum cost closed walk in $\mathcal{G}_a$, $\mathcal{G}_b$ and $\mathcal{G}_c$, given enough iterations (typically $\simeq 1000$).

The prior evidence shown in Table I is that the performance of the cyclic-strategy $\mathcal{H}TSP$ is superior to all other methods in 90% of the configurations tested. This confirms that finding a minimum cost closed walk on the graph and having robots equally spaced is usually the most effective solution for the multi-robot patrolling problem in theory. In particular, it should be noticed that in $\mathcal{G}_c$, $\eta = 1$ since all edges have the same cost of 5.70. Therefore, no other strategy was able to overcome $\mathcal{H}TSP$, which shows the potential of the approach when edges are balanced.

Furthermore, it is also important to refer that team size R plays a fundamental role when choosing a multi-robot patrolling approach. Results show that when the number of robots grows, partitioning strategies tend to approximate the performance obtained by cyclic strategies, when $\eta > 1$, as in $\mathcal{G}_a$ and $\mathcal{G}_b$. In fact, it can be seen in those cases, that when $R \geq 10$, $\mathcal{WI}_{\mathcal{EH}P} \simeq \mathcal{WI}_{\mathcal{H}TSP}$ and even $\mathcal{WI}_{\mathcal{EH}P} < \mathcal{WI}_{\mathcal{H}TSP}$. Despite only being theoretically superior to optimal cyclic-based approaches with high R, in practice a partition-based strategy like $\Pi_{\mathcal{EH}P}$ may present a great advantage over $\Pi_{\mathcal{H}TSP}$. Seeing as each robot patrols a disjoint subgraph of $\mathcal{G}$, robots do not cross the paths of each other and inter-robot coordination is not an issue. However,

[3]The degree (or valency) of a vertex of a graph is the number of edges incident to the vertex [11].

[4]Matlab code of the four methods is available at: `http://isr.uc.pt/~davidbsportugal/MRpatrol_toolbox`



(a) Environment A. (b) Environment B. (c) Environment C.

Fig. 3. Environments used in the experiments with respective topological map.

TABLE I

$\mathcal{WI}$ RESULTS (IN SECONDS), CONSIDERING UNITARY SPEED AND USING THE FOUR DESCRIBED ALGORITHMS IN THREE GRAPHS WITH DIFFERENT CONNECTIVITY AND DIFFERENT TEAM SIZES.

	Graph A ($\mathcal{G}_a$)				Graph B ($\mathcal{G}_b$)				Graph C ($\mathcal{G}_c$)			
	Cyclic		Partitioning		Cyclic		Partitioning		Cyclic		Partitioning	
R	Π_{MSTt}	$\Pi_{\mathcal{H}TSP}$	Π_{LIP}	$\Pi_{\mathcal{EHP}}$	Π_{MSTt}	$\Pi_{\mathcal{H}TSP}$	Π_{LIP}	$\Pi_{\mathcal{EHP}}$	Π_{MSTt}	$\Pi_{\mathcal{H}TSP}$	Π_{LIP}	$\Pi_{\mathcal{EHP}}$
1	516.75	**507.75**	889.80	**507.75**	380.10	**313.65**	548.10	**313.65**	273.60	**148.20**	273.60	**148.20**
2	258.37	**253.87**	441.15	273.15	190.05	**156.82**	267.90	193.35	136.80	**74.10**	136.80	79.80
3	172.25	**169.25**	294.30	178.35	126.70	**104.55**	177.00	133.20	91.20	**49.40**	91.20	57.00
4	129.19	**126.94**	215.40	141.35	95.02	**78.41**	123.90	98.10	68.40	**37.05**	68.40	45.60
5	103.35	**101.55**	175.50	107.25	76.02	**62.73**	104.10	76.20	54.72	**29.64**	45.60	34.20
6	86.12	**84.62**	144.30	100.65	63.35	**52.27**	86.40	62.40	45.60	**24.70**	45.60	34.20
7	73.82	**72.53**	120.60	83.85	54.30	**44.81**	68.10	56.70	39.09	**21.17**	34.20	22.80
8	64.59	**63.47**	104.25	65.85	47.51	**39.21**	57.60	43.50	34.20	**18.82**	34.20	22.80
9	57.42	**56.42**	96.00	61.35	42.23	**34.85**	52.20	40.20	30.40	**16.47**	22.80	22.80
10	51.67	**50.77**	84.90	52.20	38.01	**31.36**	45.90	37.20	27.36	**14.82**	22.80	22.80
11	46.98	**46.16**	76.95	47.85	34.55	**28.51**	40.20	32.10	24.87	**13.47**	22.80	22.80
12	43.06	**42.31**	67.95	45.45	31.67	**26.14**	37.20	31.80	22.80	**12.35**	22.80	22.80
13	39.75	**39.06**	62.10	39.60	29.24	**24.13**	34.20	25.50	21.05	**11.40**	**11.40**	**11.40**
14	36.91	**36.27**	59.10	39.30	27.15	**22.40**	32.70	23.10	19.54	**10.59**	11.40	11.40
15	34.45	**33.85**	53.85	36.15	25.34	**20.91**	28.80	21.90	18.24	**9.88**	11.40	11.40
16	32.29	**31.73**	48.45	36.15	23.76	19.60	25.80	**19.20**	17.10	**9.26**	11.40	11.40
17	30.39	29.87	46.35	**28.20**	22.36	**18.45**	25.50	18.60	16.09	**8.72**	11.40	11.40
18	28.71	28.21	44.40	**27.90**	21.12	17.42	23.10	**16.80**	15.20	**8.23**	11.40	11.40
19	27.19	**26.72**	41.70	27.15	20.00	16.51	21.90	**13.80**	14.40	**7.80**	11.40	11.40
20	25.84	**25.39**	40.80	26.85	19.00	15.68	21.30	**12.90**	13.68	**7.41**	11.40	11.40

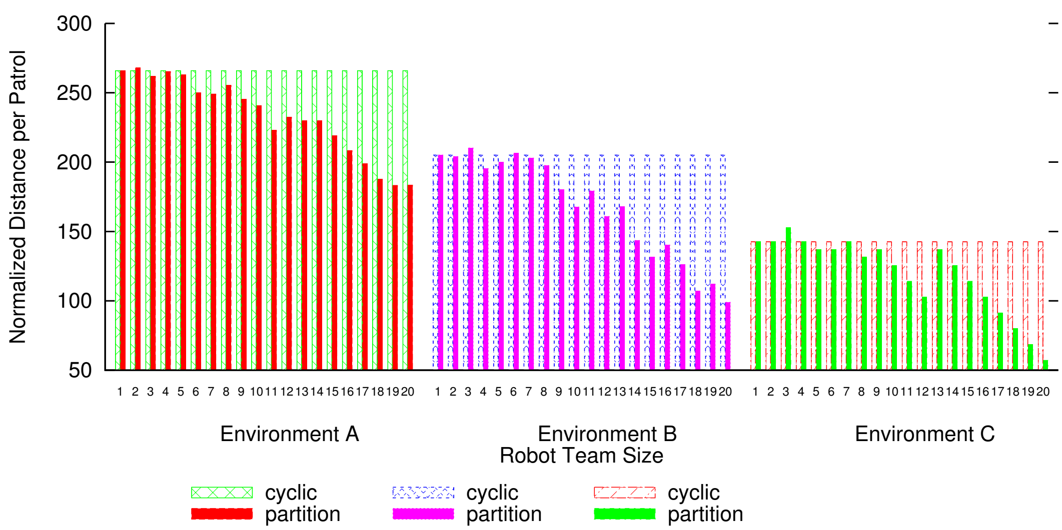


Fig. 4. Normalized distances for the cyclic ($\Pi_{\mathcal{H}TSP}$) and partition ($\Pi_{\mathcal{EHP}}$) patrol strategies for each environment. The partition strategies are more resource efficient for larger robot teams.

in a cyclic strategy, when a closed walk is computed, vertices may be repeated and robot interference is an issue. As a result, a mechanism must exist to avoid having robots visiting the same vertex at the same time. Such mechanism will have impact on the worst idleness $\mathcal{WI}_{Cyc}$ in the real world, unless the considered closed walk does not repeat vertices. The effect of robot interference has been shown previously [9].

Results also show that the heuristics considered were able to outperform methods with known performance bounds, thus reaching solutions with higher quality in all tested graphs. Additionally, despite the α-approximations reported in Theorem 1 and Remark 2, the performance of $MSTt$ and LIP was always within a factor of at most 1.85 to the best solution. Despite that, the optimal solution was only reached by LIP in one instance ($R = 13$ in $\mathcal{G}_c$). As expected, the MST-tour approximation has closer performance to $\Pi_{\mathcal{H}TSP}$ when the connectivity of $\mathcal{G}$ decreases. On the other hand, the dependency of Π_{LIP} in η is evident, seeing as its best result was obtained in $\mathcal{G}_c$.

As indicated by the results for these environments, the cyclic strategy results in lower values of $\mathcal{WI}$ in most cases. However, for longer running patrols, global resource usage of the robot team may also be an important consideration. For instance, it may be important to minimize the amount of fuel used by the robot team or the total distance covered. We expect that in the general case, the partition strategies would have lower overall resource usage than the cyclic strategies for the same environment. This is because the former are able to make graph cuts on the long edges in the graph, reducing the number of edges that must be traveled, while in the cyclic strategy, all robots must travel the entire route.

We define the *Normalized Distance per Patrol* metric to be the total distance traveled by the full robot team to perform a single patrol, divided by the average number of vertices visited on the patrol. In the cyclic case, all robots must visit each vertex at least once, while in the partition case, robots visit only the vertices in their assigned partition. The results shown in Fig. 4 indicate that the partition strategies result in lower normalized distances per patrol when the team size is greater than about 3 robots, and are therefore more resource efficient. This trend is present independently of the graph connectivity.

On a final note concerning real-world implementation, all the strategies tested rely on predefined trajectories for the robots that are computed prior to the mission start, potentially reaching an optimal solution. In some applications, this may not be intended, for example in adversarial patrolling scenarios, where an intruder may apprehend the robots' routes and attack the system in an easier way. However, in situations such as cooperative cleaning of infrastructures, having near-optimal performance is highly desired.

VII. Conclusions and Future Work

In this work, the multi-robot patrolling problem has been studied. Considering an optimization criterion based on the worst idleness, it has been shown that cyclic-based strategies tend to generate solutions with high quality and should be preferred when relatively small teams are used and/or the edge costs of the graph $\mathcal{G}$ are balanced. Partition-based strategies should be preferred when this is not the case. Cyclic strategies may result in greater overall team cost, while graph-partitioning may be desirable from a security viewpoint, use fewer resources, and require less coordination between robots. Thus, it is generally more suitable when larger teams are involved. Furthermore, the results presented show that both graph connectivity and team size play an important role on the choice of a patrolling strategy.

It was also shown that heuristic methods work well in practice, as the achieved results were superior to those of algorithms with known performance bounds. Even though we believe that the graphs used are generally representative, it would be important to conduct further tests with additional graphs. Thus, a simulation tool has been made available to the community.

Prior work on performance in multi-robot patrols [7] investigated an approach for reallocating vertices belonging to poorly performing robots. However, the design of the environment and partitioning strategy can affect the dynamic task assignment strategy. In the future, the authors plan to use the results to estimate the optimal size for a team of robots in a patrolling mission, given some imposed constraints and analyze which class of strategies should be preferred when a subset of the robotic agents do not perform as expected.

References

[1] Y. Chevaleyre, "Theoretical analysis of the multi-agent patrolling problem". In Proc. of the *2004 Int. Conf. on Agent Intelligent Technologies (IAT 2004)*, Beijing, China, September 20-24, pp. 302-308, 2004.

[2] 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, 2012.

[3] P. Fazli, A. Davoodi, P. Pasquier and A. K. MackWorth, "Complete and robust cooperative robot area coverage with limited range". In Proc. of the *Int. Conf. on Intel. Robots and Systems*, Taiwan, 2010.

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

[5] A. Machado, G. Ramalho, J. Zucker and A. Drogoul, "Multi-Agent Patrolling: an Empirical Analysis of Alternative Architectures". Lecture Notes in Comp. Science, Vol. 2581, 155-170, Springer, 2003.

[6] M. Baglietto, G. Cannata, F. Capezio and A. Sgorbissa, "Multi-Robot Uniform Frequency Coverage of Significant Locations in the Environment". In *Dist. Autonomous Robotic Systems 8*, Springer, 2009.

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

[8] A. Marino, G. Antonelli, A. P. Aguiar and A. Pascoal, "A new approach to multi-robot harbour patrolling: theory and experiments", In Proceedings of the *2012 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems* (IROS 2012), Vilamoura, Portugal, Oct. 7-12, 2012.

[9] D. Portugal and R.P. Rocha, "Distributed Multi-Robot Patrol: A scalable and fault-tolerant framework". In *Robotics and Autonomous Systems*, 61 (12), pp. 1572-1587, Elsevier, 2013.

[10] F. Pasqualetti, J. Durham and F. Bullo, "Cooperative Patrolling via Weighted Tours: Performance Analysis and Distributed Algorithms". In *IEEE Transactions on Robotics*, 28 (5), pp. 1181-1188, 2012.

[11] J. Bondy and U. Monty, "Graph Theory with Applications". Macmillan Press Ltd., 5th Edition, Elsevier, USA, 1976.

[12] J.B. Kruskal, "On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem". In Proc. of the *American Mathematical Society*, 7(1), pp. 48-50, February, 1956.

[13] G. Karypis and V. Kumar, "A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs". *Journal of Scientific Computing*, Society for Industrial and Applied Mathematics, 359-392, 20(1), 1998.

[14] M. Fiedler, "Algebraic connectivity of Graphs". In *Czechoslovak Mathematical Journal*, 23(98), 1973.