49th IEEE Conference on Decision and Control
December 15-17, 2010
Hilton Atlanta Hotel, Atlanta, GA, USA

On Optimal Cooperative Patrolling

Fabio Pasqualetti, Antonio Franchi, and Francesco Bullo

Abstract— **This work considers the problem of designing optimal multi-agent trajectories to patrol an environment. As performance criterion for optimal patrolling we consider the worst-case time gap between any two visits of the same region. We represent the area to be patrolled with a graph, and we characterize the computational complexity of the trajectory design (patrolling) problem with respect to the environment topology and to the number of robots employed in the patrolling task. Even though the patrolling problem is generally *NP-hard*, we identify particular cases that are solvable efficiently, and we describe optimal patrolling trajectories. Finally, we present a heuristic with performance guarantees, and an 8-approximation algorithm to solve the *NP-hard* patrolling problem.**

I. Introduction

The recent development in the autonomy and the capabilities of mobile robots has greatly increased the number of application suitable for a team of autonomous agents. Particular interest has been received by the tasks requiring continual execution, such as the detection of forest fires and the patrol (surveillance) of an environment [1], [2]. The surveillance of an area of interest requires the robots to continuously and repeatedly travel the environment, and the challenging problem consists in scheduling the robots trajectories so as to optimize certain performance criteria. The reader familiar with network location, multiple traveling salesman, or graph exploration problems may observe a close connection with the patrolling problem we address, e.g., see [3], [4]. It is worth noting, however, that these classical optimization problems do not capture the repetitive, and hence dynamic, aspect of the patrolling problem, nor the synchronization issues that arise when a timing among the visits of certain zones is required.

A precise formulation of the patrolling problem requires the characterization of the environment to be patrolled, of the robots capabilities, and of the performance criteria. In this work, we represent the environment as a graph, in which the vertices correspond to physical, and strategically important, locations, and in which the edges denote the possibility of moving between two connected locations. We assume that the robots are able to independently build such graph by exploring the environment. For what concern the robots, we assume them to be identical and capable of sensing and communicating within a certain spatial range, and of moving according to a first order integrator dynamic. Additionally, we assume that, when a robot is placed at each of the graph vertices, the union of the sensors footprint provide complete sensor coverage of the environment. Regarding the performance criterion, we consider the longest time gap (*refresh time*) between any two visits of the same location[1] in the environment.

The problem of designing team trajectory to satisfy certain temporal constraints has recently gained attention, e.g., see [5]. The situation in which the motion of the robots needs to guarantee a uniform frequency coverage of the environment has been considered, among others, in [6]. However, because we are interested in the worst case scenario, the trajectories proposed in these works are not optimal in our sense. In [7], an empirical evaluation of existing patrolling heuristics is performed. In [8], two classes of strategies, based respectively on space decomposition and traveling salesperson tour computation, are presented and qualitatively compared. In [2] and [9], an efficient and distributed solution to the perimeter patrolling problem is proposed.

In this work, we will study the computational complexity of the minimum refresh time patrolling problem. The computational complexity theory, a comprehensive discussion of which is in [10], is a branch of the theory of computation in computer science and mathematics that focuses on classifying problems according to their inherent difficulty. Of particular interest is the *NP-hard* class of problems, which contains all the problems that are, informally, as hard as the hardest problems in the class *NP*, for which no polynomial time algorithm is known to compute an optimal solution.

The main contributions of this work are as follows. We give a procedure to build a graph (roadmap) to represent the topological structure of the area to be patrolled, and we study the computational complexity of designing minimum refresh time team trajectories as a function of the shape of the environment, and of the number of robots employed in the patrolling task. We identify the following three cases: (i) chain environment, (ii) tree environment, and (iii) cyclic environment. For the case of a chain environment, we characterize a family of minimum refresh time team trajectories. We show that such trajectories can be computed with an algorithm which is polynomial in the number of vertices of the graph representing the environment. We also develop an algorithm to compute a partition (of given cardinality) of a chain graph, so as to minimize the maximum length of the clusters in the partition. For the case of a tree environment,

This material is based upon work supported in part by NSF grant IIS-0904501 and ARO MURI grant W911NF-05-1-0219.

Fabio Pasqualetti and Francesco Bullo are with the Center for Control, Dynamical Systems and Computation, University of California at Santa Barbara, {fabiopas,bullo}@engineering.ucsb.edu

Antonio Franchi is with the Department of Human Perception, Cognition and Action, Max Plank Institute for Biological Cybernetics, antonio.franchi@tuebingen.mpg.de

[1]Differently from the existing approaches, we leave the possibility of specifying a discrete set locations to be visited over time.

we prove that, if the number of robots is allowed to be a function of the cardinality of the graph representing the environment, then the problem of finding minimum refresh time team trajectories belongs to the class of *NP-hard* problems. However, a polynomial algorithm exists if the number of robots if fixed a priory and constant. For the latter case, we characterize a family of minimum refresh time team trajectories. Finally, we show that the general case of cyclic environment is computationally hard to solve. We propose two approximation algorithms, and we characterize their performance. The first approximate solution is extremely easy to compute, but its performance depend upon the ratio of the longest to the shortest edge in the graph representing the environment. The second approximation algorithm is based on a path-covering procedure, which is polynomial in the size of the graph, and which allows us to compute a team trajectory whose refresh time is, independently of the environment topology, within a factor 8 of the refresh time of an optimal team trajectory. To the best of our knowledge, this algorithm is the first constant factor approximation algorithm for the *NP-hard* minimum refresh time patrolling problem.

The rest of the paper is organized as follows. In Section II we define the notation and the problem under consideration. In Sections III, IV, and V we study the patrolling problem for the case of chain, tree, and cyclic environment, respectively. Finally, Section VI contains our conclusion.

II. Definitions and preliminary concepts

We are given a team of m identical robots capable of sensing and moving in a connected environment.[2]

Regarding sensing, we assume that the environment can be completely covered by simultaneously placing a robot at each of a set of n *viewpoints* in the configuration space.[3] In other words, if and only if $m = n$ robots were placed at the n viewpoints, then the union of the sensors footprint of each robot would provide complete sensor coverage of the environment. However, we assume $n > m$ so that at least one robot needs to visit more viewpoints for the entire environment to be monitored over time.

Regarding motion, we assume that the robots are holonomic, i.e., modeled as first order integrators, and move at most at unitary speed. Additionally, we associate a robotic roadmap G with the environment [12]. Precisely, the n viewpoints form the vertex set of G, and the unordered pair (i, j) belongs to the edge set of G if a robot can travel from i to j. We constrain the motion of the robots on the roadmap, and we assume that the paths corresponding to the edges of G verify the triangle inequality. Finally, we let the length of a shortest path from i to j equal the weight of the edge (i, j). An example is in Fig. 1.

Let $G = (V, E)$ denote a robotic roadmap, where V and E denote the vertex set and the edge set, respectively. A *team trajectory* X is an array of m continuous and piecewise-differentiable trajectories $x_1, \ldots, x_m$ defined by the motion of the robots on the roadmap G. We say that a viewpoint $v \in V$ is visited at time t by the robot i if $x_i(t) = v$. We define the *refresh time* of a team trajectory X, in short $\mathrm{RT}(X)$, as the longest time interval between any two consecutive visits of any viewpoint, i.e.,

$$\mathrm{RT}(X) = \max_{v \in V} \max_{(t_1,t_2) \in \Omega(v,X)} t_2 - t_1$$

where $\Omega(v, X) = \{(t_1, t_2) \,|\, x_i(t) \neq v,\ \forall i = 1, \ldots, m,\ \forall t \in (t_1, t_2)\}$.

Problem 1 (Team refresh time): Given a roadmap and a team of robots, find a minimum refresh time team trajectory.

We now present our first result on the computational complexity of designing minimm refresh time team trajectories.

Theorem 2.1 (Computational complexity of Problem 1): The *Team refresh time* problem is *NP-hard*.

Proof: Let $m = 1$, and note that a minimum refresh time trajectory consists of moving the robot at maximum speed along a shortest tour visiting the viewpoints. It follows that Problem 1 contains the minimum traveling salesman problem, which is known to be *NP-hard* [10]. By restriction [10], Problem 1 is also *NP-hard*. ■

As a consequence of Theorem 2.1, without any assumption on the input of Problem 1, the minimum refresh time optimization problem is computationally hard. In the next section, we identify two classes of input for which there exists an efficient solution to Problem 1, and later we describe two approximation algorithms to deal with the general case.

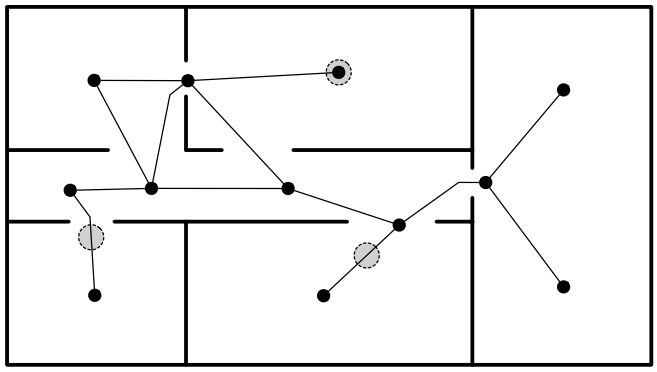

Fig. 1. A polygonal environment and an associated roadmap. The 12 vertices coincide with a set of viewpoints from which the robots provide sensor coverage of the entire environment. Each edge corresponds to the shortest path between its endpoints. The 3 robots, which are holonomic and disk-shaped (grey-filled circles), are constrained to move on the roadmap. Because of the dimension of the robots, some paths are not straight lines.

III. Minimum refresh time team trajectory on a chain roadmap

We characterize in this section an optimal refresh time team trajectory when the roadmap associated with the environment has a chain structure.

[2]Even though our results hold for any $m \in \mathbb{N}$, we focus on the case $m \geq 2$, and we remark that, if $m = 1$, then the methods developed for the shortest tour computation can be used to design patrolling trajectories.

[3]The set of viewpoints can be computed, for instance, by solving an art gallery problem for the given environment [11].

A. Open loop team trajectory characterization

Let N_i denote the neighbor set of the vertex i, and let $|N_i|$ denote the degree of i. A chain roadmap is an undirected, connected, and acyclic graph, in which every vertex has degree two, except for two vertices which have degree one. Without losing generality, we assume that the n vertices are ordered in a way that $|N_1| = |N_n| = 1$, and $N_i = \{i-1, i+1\}$ for each $i \in \{2, \dots, n-1\}$. We define a relative order of the robots according to their position on the roadmap. A team trajectory is *order invariant* if the order of the robots does not change with time, i.e., if $x_i(t) \leq x_{i+1}(t)$ for each $i \in \{1, \dots, m-1\}$ and for every instant t, where x_i denotes the distance on the roadmap from the first vertex of the chain to the position of the i-th robot.

Proposition 3.1 (Order invariant team trajectory): Let X be a team trajectory. There exists an order invariant team trajectory $\bar{X}$ such that $\text{RT}(X) = \text{RT}(\bar{X})$.

Proof: Let X be a team trajectory, and consider the permutation matrix $P(t)$, that keeps track of the order of the robots at time t. Notice that the (i,j)-th entry of $P(t)$ is 1 if, at time t, the i-th robot occupies the j-th position in the chain, and it is 0 otherwise. Since X is continuous, when the function $P(t)$ is discontinuous, the positions of the robots directly involved in the permutation overlap. Therefore, the order invariant team trajectory $\bar{X} = P^{-1}(t)X(t)$ is a feasible team trajectory such that $\text{RT}(\bar{X}) = \text{RT}(X)$. ∎

For a team trajectory X, let V_i be the set of viewpoints visited over time by the agent i, and let $\{V_1, \dots, V_m\}$ be the image of X. Let $l_i = \min_{v \in V_i} v$ and $r_i = \max_{v \in V_i} v$, and let $d_i = r_i - l_i$. Finally, let $\text{RT}^* = \min_X \text{RT}(X)$. A team trajectory is *non-overlapping* if $V_i \cap V_j = \emptyset$ for all $i, j \in \{1, \dots, m\}$ with $i \neq j$.

Proposition 3.2 (Non-overlapping team trajectory): Given a chain roadmap, there exists an order invariant and non-overlapping team trajectory with refresh time RT^*.

Proof: Let X^* be a minimum refresh time team trajectory, and let X be the order invariant team trajectory obtained from X^* as in Proposition 3.1. Clearly $\text{RT}(X) = \text{RT}^*$. Let $\{V_1, \dots, V_m\}$ be the image of X, and note that $V = \cup_{i=1}^m V_i$. Consider the partition of V defined as

$$\bar{V}_1 = V_1,$$
$$\bar{V}_i = V_i \setminus \cup_{i=1}^{i-1} V_i, \quad i = 2, \dots, m.$$

Let $\bar{l}_i = \min_{v \in \bar{V}_i} v$, $\bar{r}_i = \max_{v \in \bar{V}_i} v$, and $\bar{d}_i = \bar{r}_i - \bar{l}_i$. Note that, by construction, the viewpoint l_i is visited by the robot i and, possibly, by the robots $j > i$. Also, because X is order invariant, we have $x_i(t) \leq x_j(t)$. It follows that $\text{RT}(X) \geq 2 \max_i \bar{d}_i$. Consider now the team trajectory $\bar{X}$ with image $\{\bar{V}_1, \dots, \bar{V}_m\}$, and assume that the robots sweep periodically at maximum speed their segment. Since $\text{RT}(\bar{X}) = 2 \max_i \bar{d}_i$, the trajectory $\bar{X}$ is an order invariant and non-overlapping team trajectory with minimum refresh time. ∎

Given a chain graph on the viewpoints V, we let $\Pi_m = \{\pi_1, \dots, \pi_m\}$ denote an *m-partition* of V, such that $\pi_i \cap \pi_j = \emptyset$ whenever $i \neq j$, and $V = \bigcup_{i=1}^m \pi_i$. Additionally, we define the dimension of the partition Π_m, in short $\dim(\Pi_m)$, as the longest distance between any two viewpoints in the same cluster, i.e., as $\max_{i \in \{1,\dots,m\}} v_{\max} - v_{\min}$, where $v_{\max} = \max_{v \in \pi_i} v$ and $v_{\min} = \min_{v \in \pi_i} v$. Following Proposition 3.2, there exists a minimum refresh time team trajectory whose image coincide with an m-partition of V. Let $\dim(\Pi_m)$ denote the dimension of the m-partition Π_m. We show that the minimum refresh time equals twice the dimension of an optimal m-partition.

Trajectory 1: *Minimum refresh time trajectory on a chain roadmap (i-th robot)*

Input : l_i, r_i, $d_i = r_i - l_i$;
$x_i(t) := l_i$ for $t := 0, 2d_i, 4d_i, \dots$;
$x_i(t) := r_i$ for $t := d_i, 3d_i, 5d_i, \dots$;

Theorem 3.3: (Minimum refresh time on a chain roadmap): Let G be a chain roadmap, and let m be the number of robots. Then $\text{RT}^* = 2 \min_{\Pi_m} \dim(\Pi_m)$.

Proof: As a consequence of Propositions 3.1 and 3.2, there exists a minimum refresh time team trajectory whose image $\{V_1, \dots, V_m\}$ coincides with an m-partition Π_m. Since each robot is assigned to a different cluster, we have $\text{RT}^* \geq 2\dim(\Pi_m)$. Let X be a trajectory where each robot sweeps at maximum speed its cluster. Clearly, $\text{RT}(X) = \text{RT}^* = 2\dim(\Pi_m)$. ∎

As a consequence of Proposition 3.2 and Theorem 3.3, there exists a minimum refresh time team trajectory in which the set $\{V_1, \dots, V_m\}$ coincides with an optimal m-partition of the chain roadmap. To conclude this section, a minimum refresh time team trajectory is in Trajectory 1, where each robot sweeps at maximum speed a part of the chain graph.

B. Optimal m-partition centralized computation

In the remaining part of the section we describe an algorithm to compute an optimal m-partition.[4] Given a set of viewpoints V, we call *left-induced* partition of length l the partition $\Pi^l = \{\pi_1, \dots, \pi_k\}$ defined recursively as

$$\pi_i = \{v \in V \ : \ a_i \leq v \leq a_i + l\}, \tag{1}$$

where (cf. Fig. 2(a))

$$a_1 = v_1,$$
$$a_i = \min\{v \in V \ : \ v > a_{i-1} + l\}, \quad i = 1, \dots, k.$$

Note that, by definition, k is such that $\{v \in V \ : \ v > a_k + l\} = \emptyset$, and $\dim(\Pi^l) \leq l$. Observe that the function $|\Pi^l|$ is monotone, non-increasing, and right-continuous (cf. Fig. 2(b)). Let $\{l_1, \dots, l_{n-1}\}$ be the discontinuity points of the function $|\Pi^l|$, then, for $k \in \{1, \dots, n-1\}$,

$$|\Pi^l| \leq k, \text{ if } l \geq l_k, \quad \text{and} \quad |\Pi^l| > k, \text{ if } l < l_k. \tag{2}$$

Note that two or more discontinuity points of $|\Pi^l|$ may coincide, so that the function $|\Pi^l|$ may not assume all the values of the set $\{1, \dots, n\}$, as in Fig. 2(b) for $|\Pi^l| = 9$.

[4] By removing the longest edges in the chain the average length of the clusters is minimized. Note that, in general, such partition does not minimize the dimension of the m-partition, and hence it is not optimal in our sense.

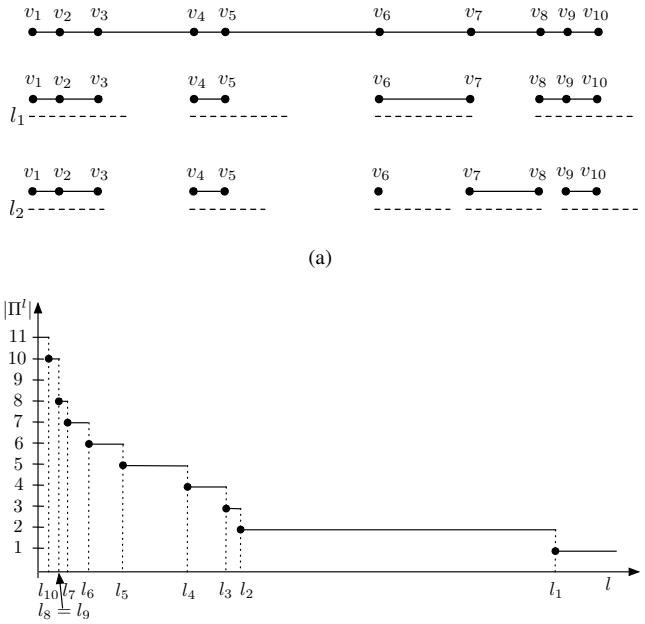

Fig. 2. In Fig. 2(a) the left-induced partition of length l_1 and l_2, with $l_2 < l_1$, for the chain roadmap with vertices $\{v_1, \ldots, v_{10}\}$. The cardinalities are $|\Pi^{l_1}| = 4$ and $|\Pi^{l_2}| = 5$, respectively. In Fig. 2(b) the cardinality $|\Pi^l|$ is plotted as a function of the length l. Notice that, because $v_2 - v_1 = v_{10} - v_9$, the function $|\Pi^l|$ does not assume the value 9.

Algorithm 2: Optimal left-induced m-partition

Input : $\{v_1, \ldots, v_n\}$, $0 < m < n$, $\varepsilon > 0$;
Set : $a := 0$, $\delta > 0$, $b := \frac{v_n + \delta}{m}$, $l := \frac{(a+b)}{2}$;
while $(b - a) > 2\varepsilon$ **do**
  $\Pi^l :=$ left-induced$(\{v_1, \ldots, v_n\}, l)$;
  if $|\Pi^l| > m$ **then**
    $a := l$, $l := \frac{a+b}{2}$;
  else
    $\Pi^* := \Pi^l$, $b := l$, $l := \frac{a+b}{2}$;
return Π^*

Theorem 3.4 (Optimal m-partition): Let G be a chain roadmap G. Let Π_m be an m-partition of G, and let Π^l be the left-induced partition of length l of G. Then

$$\min_{\Pi_m} \dim(\Pi_m) = \min\{l : |\Pi^l| \leq m\}.$$

Proof: Let Π_m be an m-partition, and let $\Pi^l = \{\pi_1^l, \ldots, \pi_k^l\}$ be the left induced partition of length l of a chain roadmap G. Let $l^* = \min_{\Pi_m} \dim(\Pi_m)$. We want to show that l^* is one of the discontinuity points of the function $|\Pi^l|$, i.e., that l^* verifies the conditions (2). By contradiction, if $l < l^*$ and $|\Pi^l| \leq m$, then an m-partition with dimension smaller than the optimal would exists. Therefore, if $l < l^*$, then $|\Pi^l| > m$. Suppose now that $l \geq l^*$, and let $\Pi_m^* = \{\pi_1^*, \ldots, \pi_m^*\}$ be an m-partition with minimum dimension. Notice that $|\pi_1^l| \geq |\pi_1^*|$, because the cluster π_1^l contains all the viewpoints within distance l from v_1, and hence also within distance l^*. It follows that $\max \pi_1^l \geq \max \pi_1^*$, and also that $\min \pi_2^l \geq \min \pi_2^*$. By repeating the same procedure to the remaining clusters, we obtain that $\max \pi_m^l \geq \max \pi_m^*$, so that, if $|\Pi^*| = m$ and $l \geq l^*$, then $|\Pi^l| \leq m$. ■

Following Theorem 3.4, an optimal left-induced partition of cardinality at most m is also an optimal m-partition. A procedure to compute an optimal left-induced partition is in Algorithm 2, where the function *left-induced($\{v_1, \ldots, v_n\}$,l)* returns the left-induced partition defined in (1).[5] We next characterize its convergence properties.

Lemma 3.5 (Convergence of Algorithm 2): Let G be a chain roadmap, and let Π_m be an m-partition of G. Let $l^* = \min_{\Pi_m} \dim(\Pi_m)$. Then,

(i) Algorithm 2 returns a left-induced partition of dimension at most $l^* + \varepsilon$, and cardinality at most m, and
(ii) the time complexity of Algorithm 2 is $O(n \log(\varepsilon))$.

Proof: Algorithm 2 search for the minimum length l^* that generates a left-induced partition of cardinality at most m. Because of Theorem 3.4, the length l^* coincides with one of the discontinuity points of the function $|\Pi^l|$, and it holds $l^* \in (0, v_n/m + \delta)$, where $\delta > 0$. Indeed, $l^* > 0$ because $m < n$, and $l^* < v_n/m + \delta$, because $(v_n/m + \delta)m > v_n$. Recall from (2) that $|\Pi^l| > m$ for every $l < l^*$, and that the function $|\Pi^l|$ is monotone. Note that the interval $[a, b]$, as updated in Algorithm 2, contains the value l^* at every iteration. The length of the interval $[a, b]$ is divided by 2 at each iteration, so that, after $\log_2((v_n/m + \delta)/\varepsilon)$, the value l^* is computed with precision ε. Since the computation of $|\Pi^l|$ can be performed in $O(n)$ operations, the time complexity of Algorithm 2 is $O(n \log(\varepsilon))$. ■

For ease of notation, in the following sections, we use the set $\{V_1, \ldots, V_m\}$ to denote both the image of a team trajectory and an m-partition of the chain graph.

IV. Minimum refresh time team trajectory on an acyclic roadmap

The problem of designing team trajectories for a tree roadmap is now considered. Let $T = (V, E)$ denote an undirected, connected, and acyclic roadmap (tree). Recall that a vertex path is a sequence of vertices such that any pair of consecutive vertices in the sequence are adjacent. A tour is a vertex path in which the start and end vertices coincide, and in which every vertex of T appears at least once in the sequence. A depth-first tour of T is a tour that visits the vertices V in a depth-first order [13]. Let DFT(T) denote the length of a depth first tour of T. Notice that the length of a depth-first tour of a connected tree equals twice the sum of the length of the edges of the tree, and that any depth-first tour is a shortest tour visiting all the vertices. We now show that, for the case of tree roadmap, the set of cyclic and partition strategies described in [8] does not contain, in general, a minimum refresh time trajectory. Recall that in a cyclic based strategy the robots travel at maximum speed and equally spaced along a minimum length tour visiting all the viewpoints. Consider the tree roadmap of Fig. 3(a), and

[5] A distributed implementation of Algorithm 2 requires the computation of the left-induced partition in a distributed way. Such computation can be performed by simple programming operations and it is not described here.

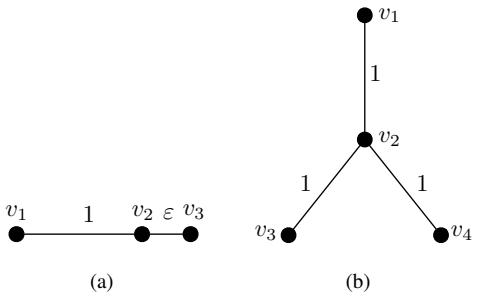

Fig. 3. Two examples of tree roadmap. For these topologies, if two robots are assigned to the patrolling task, both the cyclic based strategy and the partition based strategy do not provide minimum refresh time.

suppose that two robots are assigned to the patrolling task. Clearly, the minimum refresh time is 2ε, while the refresh time of a cyclic strategy equal $1+\varepsilon$. Consider now the tree roadmap in Fig. 3(b), where the edges have unit length, and assume that two robots are in charge of the patrolling task. Observe that any partition of cardinality 2 contains a chain of length 2, so that, since only one robot is assigned to each cluster, the minimum refresh time that can be obtained is 4. Suppose, instead, that the robots visit the vertices of the roadmap as specified in Table I, where $x(t)$ denotes the position of a robot at time t. Since the refresh time of the proposed trajectory is 3, we conclude that neither the cyclic based nor the partition based strategy may lead to a minimum refresh time team trajectory on a tree roadmap.

TABLE I

Robot	$x(0)$	$x(1)$	$x(2)$	$x(3)$	$x(4)$	$x(5)$	$x(6)$	$x(7)$
1	v_1	v_2	v_4	v_2	v_3	v_2	v_1	$\cdots$
2	v_2	v_3	v_2	v_1	v_2	v_4	v_2	$\cdots$

We now introduce some definitions. Let X be a team trajectory on the tree roadmap T. We say that the edge $(v_j, v_k) \in E$ is used by X if there exists $i \in \{1, \dots, m\}$ and $t_1, t_2 \in [0, \text{RT}(X)]$ such that $x_i(t_1) = v_j$ and $x_i(t_2) = v_k$, and it is unused otherwise. Note that, because in a tree there exists only one path connecting two vertices, the above condition ensures that the edge (j,k) is traveled by the robot i. Let $\bar{E}$ denote the set of unused edges, and let F_T be the forest obtained from T by removing the edges $\bar{E}$ from E, i.e., the collection of subtrees $\{T_1, \dots, T_k\}$, with $T_i = (V_i, E_i)$, such that $V = \cup_{i=1}^k V_i$ and $E_i \subseteq E$, for each $i \in \{1, \dots, k\}$. Let m_i be the number of robots that visit at least one vertex of T_i in the interval $[0, \text{RT(X)}]$, and note that $m_i > 0$ because we consider finite refresh time trajectories. Let $M = \{m_1, \dots, m_k\}$. We refer to the pair (F_T, M) as the subtree collection associated with the team trajectory X. Notice that the same subtree collection can be associated with different team trajectories. We say that a team trajectory is *efficient* if its refresh time is the smallest among all the team trajectories associated with the same subtree collection.

Theorem 4.1 (Efficient team trajectory): Let (F_T, M) be the subtree collection associated with the team trajectory X on the tree roadmap T, where $F_T = \{T_1, \dots, T_k\}$, and $M = \{m_1, \dots, m_k\}$. Then, X is efficient if

$$\text{RT}(X) = \max_{j \in \{1,\dots,k\}} \text{DFT}(T_j)/m_j.$$

Proof: Let $i \in \{1, \dots, k\}$, and let m_i be the number of robots assigned to T_i. Notice that the robots in T_i travel, in total, at least $\text{DFT}(T_i)$ to visit all the vertices. Since the speed of the robots is bounded by 1, the smallest refresh time for the vertices of T_i is $\text{DFT}(T_i)/m_i$. ∎

Given a subtree collection, an efficient team trajectory, an example of which is in Table I, can be computed with the following procedure.

Lemma 4.2 (Efficient team trajectory computation): Let (F_T, M) be a subtree collection of a tree roadmap, where $F_T = \{T_1, \dots, T_k\}$, and $M = \{m_1, \dots, m_k\}$. An efficient team trajectory is as follows: for each $i \in \{1, \dots, k\}$,

(i) compute a depth-first tour τ_i of T_i,
(ii) equally space m_i robots along τ_i, and
(iii) move the robots clockwise at maximum speed on τ_i.

Proof: Let X be the team trajectory defined above, and observe that, for each $T_j \in F_T$, the leaves of T_j appear only once in a depth first tour of T_j. It follows that the leaves are visited every $\text{DFT}(T_j)$ instants of time. Let v be a vertex of T_j, and let v have degree d. Note that v appears d times in a depth first tour of T_j. Clearly v is visited at least every $\text{DFT}(T_j)/m_j$, so that $\text{RT}(X) = \max_{j \in \{1,\dots,k\}} \text{DFT}(T_j)/m_j$. ∎

Let $P(m)$ be the partition set of m, i.e., the set of all the sequences of integers whose sum is m. The following problem is useful to characterize the complexity of designing minimum refresh time trajectories on a tree roadmap.

Problem 2 (Optimal subtree collection): Let T be a tree roadmap and m the number of robots. Find a subtree collection (F_T, M) that minimizes $\max_{j \in \{1,\dots,|F_T|\}} \text{DFT}(T_j)/m_j$ subject to $M \in P(m)$ and $|F_T| = |M|$.

Lemma 4.3 (Equivalent problem): For the case of a tree roadmap, Problems 1 and 2 are equivalent.

Proof: As a consequence of Theorem 4.1, the minimum refresh time on a tree roadmap T can be written as $\min_{(F_T,M)} \max_{j \in \{1,\dots,k\}} \text{DFT}(T_j)/m_j$, where (F, M) is a subtree collection of T, and $|M| = |F_T| = k \leq m$. It follows that a solution to Problem 1 can be derived in polynomial time from a solution to Problem 2 by using the procedure described in Lemma 4.2. Suppose now we have a solution to Problem 1, then an optimal subtree collection follows from the identification of the unused edges. We conclude that the two optimization problems are equivalent. ∎

We are now able to prove the computational complexity of the problem of designing minimum refresh time team trajectories on a tree roadmap.

Theorem 4.4: (Computational complexity of designing minimum refresh time trajectories on a tree): Let T be a tree roadmap, and let m be the number of robots. The optimization Problem 1 with input (T, m) is *NP-hard*.

Proof: By Theorem 4.3, Problem 1 is equivalent to Problem 2. Let $m = 2$, and $P(m) = \{\{2\}, \{1,1\}\}$. Then Problem 2 contains the *NP-hard* problem presented in [14], and it is therefore also computationally hard. ∎

Remark 1 (Fixed number of robots): If both the tree roadmap T and the number of robots m are arbitrary, then, because of Theorem 4.4, the problem of designing minimum refresh time team trajectories is computationally hard. However, if the number of robots is not part of the input of Problem 2, then this optimization problem is solvable in polynomial time, and precisely with time complexity $O((m-1)!n)$. We refer the interested reader to [15] for a detailed proof. In this situation, Lemma 4.2 can be used to efficiently compute a minimum refresh time team trajectory.

V. Minimum refresh time team trajectory on a cyclic roadmap

In this section we propose an approximate solution to Problem 1 in the case of a cyclic roadmap. Let $G = (V, E)$, with $|V| = n$, be an undirected and connected roadmap. Note that there exists an open tour τ with $2(n-1)$ edges that visits all the vertices.[6] We associate a chain roadmap Γ with τ, such that Γ has $2n-1$ vertices and $2(n-1)$ edges, and such that the length of the i-th edge of Γ equals the length of the i-th edge of τ. Our first approximation method consists of computing Trajectory 1 for an optimal m-partition of Γ.

Theorem 5.1: Performance of the chain approximation strategy): Let G be a connected roadmap, let n be the number of vertices of G, and let δ be the ratio of the longest to the shortest length of the edges of G. Let RT^* be the minimum refresh time on G. Let τ be an open tour with $2(n-1)$ edges that visits all the n vertices, and let Γ be the chain roadmap associated with τ. Let RT^*_Γ be the minimum refresh time on Γ. Then $\mathrm{RT}^*_\Gamma \leq 8\delta \mathrm{RT}^*$.

Proof: Let $\underline{w}$ be the shortest length of the edges of G, and note that the length of Γ is upper bounded by $2n\delta\underline{w}$. It follows that $\mathrm{RT} \leq \frac{4n\delta\underline{w}}{m}$. Since $m < n$ by assumption, some robots need to move along G for all the viewpoints to be visited. Because each robot can visit only a vertex at a time, at least $\left\lceil \frac{n}{m} - 1 \right\rceil$ steps are needed to visit all the vertices of G, and therefore $\mathrm{RT}^* \geq \left\lceil \frac{n}{m} - 1 \right\rceil \underline{w} \geq \frac{1}{2}\frac{n}{m}\underline{w}$. By taking the ratio of the two quantities we get $\mathrm{RT}^*_\Gamma \leq 8\delta \mathrm{RT}^*$. ■

In what follows, we describe a polynomial time constant factor approximation algorithm for the Team refresh time problem. Given a roadmap $G = (V, E)$ and a positive integer $k < |V|$, we define a path cover of cardinality k as the collection of paths $\{p_1, \ldots, p_k\}$ such that $V \subseteq \bigcup_{i=1}^{k} p_i$. Let the cost of a path equal the sum of the length of its edges. The min-max path cover problem asks for a minimum cost path cover for the input graph, where the cost of a cover equals the maximum cost of a path in the cover.

Theorem 5.2 (Min-max path cover [4]): There exists a 4-approximation polynomial algorithm for the *NP-hard* min-max path cover problem.

A constant factor approximation algorithm for the *NP-hard* Problem 1 is described in the proof of the next theorem.

Lemma 5.3 (Constant factor approximation): There exists an 8-approximation polynomial algorithm for the *NP-hard* Problem 1.

[6] An open tour with $2(n-1)$ edges that visit all the vertices is constructed starting from a spanning tree of G.

Proof: Let $\{p_1, \ldots, p_m\}$ be a 4-approximation path cover of the graph G. Note that the length of each path is within $4\mathrm{RT}^*$. Indeed, in a minimum refresh time team trajectory, every vertex is visited after at most RT^* instants of time. Let X the team trajectory obtained by letting each robot sweep at maximum speed a different path. Clearly, $\mathrm{RT}(X) \leq 8\mathrm{RT}^*$. Notice that, because of Theorem 5.2, the team trajectory X can be computed in polynomial time. ■

VI. Conclusion

The problem of designing the trajectory of a team of robots to patrol an environment has been considered. With respect to the refresh time criterion, optimal team trajectories have been described. The computational complexity of the design problem has been characterized as a function of the environment topology and of the cardinality of the team to be employed. It is shown that the patrolling problem is generally *NP-hard*, and a polynomial time 8-approximation algorithm has been proposed. Particular instances have been identified for which an exact optimal solution can be computed efficiently.

References

[1] D. W. Casbeer, D. B. Kingston, R. W. Beard, T. W. Mclain, S.-M. Li, and R. Mehra, "Cooperative forest fire surveillance using a team of small unmanned air vehicles," *International Journal of Systems Sciences*, vol. 37, no. 6, pp. 351–360, 2006.

[2] Y. Elmaliach, A. Shiloni, and G. A. Kaminka, "A realistic model of frequency-based multi-robot polyline patrolling," in *International Conference on Autonomous Agents*, Estoril, Portugal, May 2008, pp. 63–70.

[3] B. C. Tansel, R. L. Francis, and T. J. Lowe, "Location on networks: a survey. Part I: the p-center and p-median problems," *Management Science*, vol. 29, no. 4, pp. 482–497, 1983.

[4] E. M. Arkin, R. Hassin, and A. Levin, "Approximations for minimum and min-max vehicle routing problems," *Journal of Algorithms*, vol. 59, no. 1, pp. 1–18, 2006.

[5] F. Amigoni, N. Basilico, and N. Gatti, "Finding the optimal strategies for robotic patrolling with adversaries in topologically-represented environments," in *IEEE Int. Conf. on Robotics and Automation*, Kobe, Japan, May 2009, pp. 2005–2010.

[6] Y. Elmaliach, N. Agmon, and G. A. Kaminka, "Multi-robot area patrol under frequency constraints," in *IEEE Int. Conf. on Robotics and Automation*, Roma, Italy, Apr. 2007, pp. 385–390.

[7] A. Machado, G. Ramalho, J. D. Zucker, and A. Drogoul, "Multi-agent patrolling: An empirical analysis of alternative architectures," in *Multi-Agent-Based Simulation II*, ser. Lecture Notes in Computer Science. Springer, 2003, pp. 155–170.

[8] Y. Chevaleyre, "Theoretical analysis of the multi-agent patrolling problem," in *IEEE/WIC/ACM Int. Conf. Intelligent Agent Technology*, Beijing, China, Sep. 2004, pp. 302–308.

[9] D. B. Kingston, R. S. Holt, R. W. Beard, T. W. McLain, and D. W. Casbeer, "Decentralized perimeter surveillance using a team of UAVs," in *AIAA Conf. on Guidance, Navigation and Control*, San Francisco, CA, Aug. 2005.

[10] M. R. Garey and D. S. Johnson, *Computers and Intractability*. Springer, 1979.

[11] A. Ganguli, J. Cortés, and F. Bullo, "Distributed deployment of asynchronous guards in art galleries," in *American Control Conference*, Minneapolis, MN, Jun. 2006, pp. 1416–1421.

[12] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006, available at `http://planning.cs.uiuc.edu`.

[13] D. Peleg, *Distributed Computing. A Locality-Sensitive Approach*, ser. Monographs on Discrete Mathematics and Applications. SIAM, 2000.

[14] I. Averbakh and O. Berman, "A heuristic with worst-case analysis for minimax routing of two travelling salesmen on a tree," *Discrete Applied Mathematics*, vol. 68, no. 1-2, pp. 17–32, 1996.

[15] H. Nagamochi and K. Okada, "A faster 2-approximation algorithm for the minmax p-traveling salesmen problem on a tree," *Discrete Applied Mathematics*, vol. 140, no. 1-3, pp. 103–114, 2004.