Journal of Intelligent & Robotic Systems manuscript No.
(will be inserted by the editor)

DARP: Divide Areas Algorithm For Optimal Multi-Robot Coverage Path Planning

Athanasios Ch. Kapoutsis · Savvas A. Chatzichristofis · Elias B. Kosmatopoulos

Received: date / Accepted: date

Abstract This paper deals with the path planning problem of a team of mobile robots, in order to cover an area of interest, with prior-defined obstacles. For the single robot case, also known as single robot coverage path planning (CPP), an $\mathcal{O}(n)$ optimal methodology has already been proposed and evaluated in the literature, where n is the grid size. The majority of existing algorithms for the multi robot case (mCPP), utilize the aforementioned algorithm. Due to the complexity, however, of the mCPP, the best the existing mCPP algorithms can perform is at most 16 times the optimal solution, in terms of time needed for the robot team to accomplish the coverage task, while the time required for calculating the solution is polynomial. In the present paper, we propose a new algorithm which converges to the optimal solution, at least in cases where one exists. The proposed technique transforms the original integer programming problem (mCPP) into several single-robot problems (CPP), the solutions of which constitute the optimal mCPP solution, alleviating the original mCPP explosive combinatorial complexity. Although it is not possible to analytically derive bounds regarding the complexity of the proposed algorithm, extensive numerical analysis indicates that the complexity is bounded by polynomial curves for practical sized inputs. In the heart of the proposed approach lies the DARP algorithm, which divides the terrain into a number of equal areas each corresponding to a specific robot, so as to guarantee *complete coverage*, *non-backtracking solution*, *minimum coverage path*, while at the same time *does not need any preparatory stage* (video demonstration and standalone application are available on-line http://tinyurl.com/DARP-app).

Keywords mCPP, multi-robots, complete coverage, minimum coverage paths, terrain sub-division.

A. Ch. Kapoutsis, S. A. Chatzichristofis and E. B. Kosmatopoulos are with the Department of Electrical and Computer Engineering, Democritus University of Thrace, Xanthi Greece & Information Technologies Institute, CERTH Thessaloniki, Greece e-mail: athakapo@iti.gr.

1 Introduction

Since the 1970s, autonomous robots have been in daily use at very low and very high altitudes, for deep-sea and space exploration and in almost all aircrafts [20]. Today, in the era of multi robots, many of the robotic challenges, with a definite solution for the case of single robot, have to be revised so as to optimally incorporate the multi-robot dynamics. One of the fundamental problems in robotics is to determine an optimal path involving all points of a given area of interest, while avoiding sub-areas with specific characteristics (e.g., obstacles, no-fly zones, etc.). In the literature, this problem is often refereed to as coverage path planning problem (CPP), but can also be found as sweeping, exhaustive geographical search, area patrolling etc. This task is directly related with a plethora of robotic applications, such as vacuum cleaning robots [1],[26], autonomous underwater vehicles [23],[22], unmanned aerial vehicles [31], demining robots [4], automated harvesters [28], planetary exploration [6], search and rescue operations[34].

The usual abstraction of the problem, consists of a robot with an associated *tool* (e.g. sensor, actuator), which is able to spatially cover at least the size of the robot itself. Therefore, one of the most common area representation techniques is to separate the field into identical cells (e.g. in the size of robot), such that the coverage of each cell can be easily achieved. Apparently, for any arbitrary shaped area, the union of the cells only

approximates the target region, thus this technique, which is also adopted in our approach (see section 3), is termed as *approximate cellular decomposition*. A comprehensive analysis of the different area decomposition techniques along with the major representatives from each class can be found in [11].

During the previous decade, researchers focus their effort to the aforementioned *single robot* coverage planning problem (inside an already known terrain), producing a lot of different approaches (e.g. [10],[38],[36]). One of the dominant approaches is the spanning tree coverage (STC) algorithm [18], which is able to guarantee an optimal covering path in linear time, constructing a minimum spanning tree for all the free cells. The term *optimal* encapsulates that, the generated path does not revisit the same cell (*non-backtracking* property), *completely covers* the area of interest and it achieves all the above *without any preparatory effort* (the robot can be initiated at any non-occupied cell). This major accomplishment comes with the assumption that the operation area does not get "more narrow" than the double of the robot's size. Our approach utilizes the STC algorithm, thus it inherits this requirement, which is more formally described in the section 4 of the paper.

The recent advances in robotics technology, both in the hardware and in the associated software, expand the variety of robots that can be deployed for a coverage task. As a consequence, the usage of multi-robots teams in the coverage path planning problem (forming the multi-CPP or mCPP problem), has recently received a lot of attention. Unfortunately, the mCPP problem was proven to be extremely more difficult to be adequately addressed. As a matter of a fact, solving mCPP with the minimal covering time has been proven to be NP-hard [39]. Previous investigations attempt to overcome the NP nature of the problem by proposing algorithms that solve a relaxed version of the original mCPP problem, mostly focusing only on one of the main coverage objectives (see section 2 for more details). Moreover, in the mCPP problem, besides the optimality features that characterize a solution and derived directly from the single-CPP, the challenge to design the paths in a way to *fully exploit the available multi-robot dynamics* arises. In essence, this condition is one of the holy grails in any multi-robot system, since the unlock of such a feature would allow the fully cooperation of the robots with the ultimate utilization of their capabilities. In many of the proposed approaches, the fully exploitation of multi-robots dynamics is sacrificed for the sake of the main coverage objective (completeness, non-backtracking). Additionally, in multi-robot approaches, an often omitting issue is the needed cost/time in order to "transfer" the robots in their starting cells, excluding the initial robots location from the problem. Overall, the best of the proposed approaches can achieve coverage time which can be 16 times greater than the optimal one, in strictly polynomial time.

In the present paper we propose a methodology that is able to deliver the optimal solution for the mCPP problem - at least where one exists- in terms of coverage time, without overlooking any of the aforementioned aspects. In contrary to the traditional addressing of this problem [14] (usually referred as *allocate-then-decompose* or *decompose-then-allocate*), where the building and allocation of the tasks are tackled in a separated fashion [29], a new method in which the building task is *robot-oriented* is presented. Simultaneously, extended numerical analysis in realistic environments indicates that the computational time is polynomial in the size of grid times the #robots. In essence, the original mCPP is transformed into an optimization problem, where the satisfaction of a well-defined set of constraints will eventually give rise to the optimal solution. More precisely, the proposed scheme is separated into two phases.

- First, the available cells are divided into distinct classes, as many as the #robots, by utilizing a constraint satisfaction scheme. The aim of this clustering is to preserve the following attributes a) the *complete coverage*, b) the operation *without any preparatory effort* and - most importantly - c) the *fully exploitation of multi-robots dynamics*. In the heart of the proposed algorithm, lies the Divide Areas based on Robot's initial Positions (DARP) algorithm which is able to produce the optimal cells assignment with respect to the initial positions of the robots. The later can be achieved by employing a - specifically tailored to the problem at hand - cyclic coordinate descent approach [35] with the known convergence properties.
- During the second phase, the STC algorithm designs the optimal path for every robot's cluster, in a distributed manner.

The outline of the paper is as follows. The related work is described in section 2, presenting alternative works on the mCPP. The mCPP problem is transformed into an optimization problem in section 3, introducing all the essential notation. In section 4 are briefly summarized the main steps of the STC algorithm, regarding the optimal solution of CPP problem. The findings of that section are going to be utilized in order to relax the original mCPP problem in section 5. On the same section, are formally described the essential conditions of the optimal solution. In section 6 is proposed the DARP algorithm, with a comprehensive discussion about its performance. The complete scheme for the

mCPP problem is outlined in section 7. As proof of concept, in section 8 is presented the performance of the proposed scheme in comparison with two of the state-of-the-art algorithms, regarding the mCPP problem. Finally, the concluding remarks are drawn in section 9 together with an outlook to the future work.

2 Related Work

2.1 Multi-Robot Coverage Path Planning Problem inside known terrain

Despite the fact that mCPP is a relative young field of research, there is a plethora of works that attempt to address the limitations and the restrictions of this problem. An in-depth discussion of this field is beyond the scope of this paper, thus, in oder to construct a more appropriate and homogeneous pool of alternative works, only publications that are in line with our problem formulation (section 3) are included. For a more detailed and complete survey with regards to the latest achievements on the CPP/mCPP problem the reader should refer to [19].

The authors in [21] transformed, for the first time, the single robot Spanning Tree Coverage (STC) Algorithm [18] into a method that is able to incorporate team of robots. Their centralized algorithm (referred as MSTC) guarantees the *complete coverage* of the operational area while avoids a-priori known obstacles. Moreover, the *non-backtracking* version produces a solution that visits every cell only once, while it is robust to robot's failures. Unfortunately, the path length for each robot is critically depended on the initial position of the robots and indeed in the worst case scenario, the maximum path length for the one robot is almost equivalent to that of a single robot case, even though there may exist alternative optimal paths configurations.

The same authors, in an attempt to alleviate the aforementioned shortcoming, proposed an enhanced version (referred as OPT-MSTC) [5], in which the form of the spanning tree is modified so as to minimize the maximum distance between every two consecutive robots along the spanning tree path. This technique performs statistically better than the random generated tree, but again without any guarantee with respect to the initial robots' positions.

An alternative technique that also utilizes spanning trees, was presented in [39]. In this work, the authors provide an upper bound on the performance of a multi-robot coverage algorithm on known terrain, guaranteeing a performance *at most sixteen times the optimal cost*, preserving at the same time the key feature of *complete coverage*. Although the *non-backtracking* guarantee has been now removed, the MFC algorithm performs significantly better from both MSTC and OPT-MSTC in terms of minimizing the maximum robot's path length, revealing that solutions without the equality constraint in the robot's path length are far away from the optimal team utilization.

The authors in [17], developed a methodology that attempts to solve the problem of patrolling a known environment by a team of mobile robots, which can be translated to visiting all the points of the terrain with a certain frequency. Indeed, the patrol problem is closely related to the mCPP problem, therefore solutions that are used for patrolling might be used for mCPP as well. In this work, the authors first produce a minimal cyclic path, similar to [18], that traverse every single cell of the operation area and afterwards they search for the best "new" robots initial positions. These new locations are calculated so as to minimize the maximum distance from their initial positions and these to-be-traveled distances to be more or less the same. Unfortunately, this separation into two independent tasks, restricts the performance of the proposed algorithm. As a matter of fact, there is no upper bound regarding of number of the cells that are going to be visited in the worst case scenario, in order to fulfill the condition about the equality in robots' paths, even in cases where an alternative optimal solution actually exists.

2.2 Area division, for multi-robot tasks

This subsection presents the dominant area division techniques, in order to assist multi-robot tasks - not limited to coverage.

An interesting method that falls in this class, has been presented in [25]. The operation area is divided using sweep-line approach and in the sequel, each subarea is assigned to the most appropriate robot, based on their relative capabilities. However, the approach assumes as essential the unrealistic condition that the robots are initially located on the boundaries of the operation area. Moreover, the presented algorithm considers only convex areas without obstacles.

In [8], the authors proposed a complete approach for multi-UAV area coverage problem with a direct application to the task of remote sensing in agriculture. As first step, the authors proposed an area subdivision method, which expands the well-known *alternate-offer* protocol [30]. This technique, aims to perform the tasks of area division and assignment simultaneously, but in a distributed effective way. Despite the well establishment of the method in terms of implementation details, there is no performance guarantee. The authors state

that the final subareas assignment is a perfect equilibrium, but there is no reference on how the approach overcomes sub-optimal cases, which will be inevitably appeared in cases of non-convex areas or "difficult" initial robots' placement.

The authors in [3], presented an alternative method using a heuristic algorithm to tackle the problem of arbitrary polygon division. Despite the fact that the results are rather promising and their algorithm runs in polynomial time, the produced solution has two main disadvantages. On the one hand, there is no specific guarantee about the optimality of the area division, while at the other hand the initial robots' positions are not taking into account.

The algorithm described in [29], aims to achieve an enhanced multi-robot exploration by dividing workplace into separated regions for each available robot. The authors, by employing the K-means algorithm, divide the available terrain into distance-related, convex subregions and afterwards apply a robot-subregion assignment mechanism to the transformed linear programming problem, utilizing LP-solve software [2]. Unfortunately, this two stage procedure may end up with highly sub-optimal solutions, where it might be required for the robots to travel long distances (in comparison to the whole operation area) in order to reach their assigned subareas.

Many of the state-of-the-art approaches regarding the area division problem in multi robot context (e.g. [9],[13],[16]), have been relied on the Lloyd's [24] algorithm, with the known convergence properties [15], and/or the Voronoi partitioning [7]. Although, these approaches seem suitable for the mCPP problem, and especially for the area division problem, they differ at a quite important aspect. These approaches seek to answer the following question: "Which are the most preferable positions to place the robots, so as to cover the non-occupied space with their on-board sensors?" On the contrary, in the present paper the term "cover" implies that the respective robot has to physically visit the corresponding assigned area. The aforementioned approaches are better suited for problems, such as to position a team of robots in a terrain so that any location is as close as possible to at least one robot [12] or to optimally monitoring a dynamic event with heterogeneous sensory interest (e.g. oil spill) [32]. Thus, the majority of these approaches solves the area division problem independently of the robots/agents initial positions. Therefore, the direct appliance of these algorithms to the mCPP problem may lead to quite sub-optimal results as the robots' areas may be equally divided, but the time/cost to reach these sub-areas has been left out of the equation.

The fine-grained analysis of the related literature clearly indicates that there is room for contributions, so as to enhance the fully exploitation of the robots capabilities without jeopardizing important features of the already produced solutions. According to this necessity, this work proposes a grid-based multi-robot path planning algorithm inside known terrains, performing an area subdivision, according to both the number of robots and to their initial locations. In a subsequent stage, the exact paths inside each robot-exclusive area are defined in a completely distributed manner. The proposed algorithm is an approximate polynomial-time algorithm (for practical sized inputs) for the mCPP problem, which is able to guarantee that the solution i) *complete covers all the area* ii) *without backtracking in already visited sub-areas* iii) *guarantees the minimum coverage time* exploiting all the available robots iv) *and does not need any preparatory stage* (the robots can start their journeys from their initial positions).

3 Multi-Robot Coverage Path Planning Formulation

For ease of understanding, it is assumed that the terrain to be covered is constrained within a rectangle[1] in the (x, y)-coordinates and it is discretized into finite set of equal cells, the number of which represent both the level of required spatial resolution and the sensing capabilities of the robots.

$$\mathcal{U} = \{x, y : x \in [1, rows], y \in [1, cols]\} \tag{1}$$

where $rows, cols$ are the number of rows and columns after the discretization of the terrain to be covered. Apparently, the number of all the terrain's cells is given by $n = rows \times cols$.

It is also assumed that there are n_o obstacles placed in a-priori known positions of $\mathcal{U}$. The set of unknown obstacles is represented as:

$$B = \{(x, y) \in \mathcal{U} : (x, y) \text{ is } occupied\} \tag{2}$$

Robots cannot traverse obstacles, thus the overall set of cells that need to be covered is reduced to:

$$\mathcal{L} = \mathcal{U} \setminus B \tag{3}$$

and the number of cells to be covered is reduced to $l = n - n_o$

Definition 1 Two cells (x_i, y_i) and (x_j, y_j) are considered *adjacent* if:

$$\|x_i - x_j\| + \|y_i - y_j\| \leq 1 \tag{4}$$

[1] However, the problem formulation along with the proposed algorithm could be straightforwardly applied to different area shapes, not necessarily convex

As typical in many multi-robot coverage approaches, it is assumed that the robot can perfectly localize itself inside $\mathcal{U}$ and at each time-stamp, it can travel from its current cell to any unblocked ($\in \mathcal{L}$) *adjacent* cell, without any motion uncertainty.

Definition 2 As valid robot *path* of length m is considered every sequence of cells

$$X = ((x_1, y_1), \ldots, (x_m, y_m))$$

where the following constraints are hold

- $(x_i, y_i) \in \mathcal{L}$, $\forall\, i \in \{1, \ldots, m\}$
- every two sequential cells, i.e. (x_i, y_i) and (x_{i+1}, y_{i+1}), are *adjacent* (Definition 1), $\forall i \in \{1, \ldots, m-1\}$.

Moreover, a *closed path* of length m is a *path*, as defined in Definition 2, where the additional condition is hold

- (x_1, y_1) and (x_m, y_m) are *adjacent*

The robot positions are defined as:

$$\chi_i(t) = (x_i, y_i) \in \mathcal{L},\ \forall i \in \{1, \ldots, n_r\} \tag{5}$$

where t denotes the specific time-stamp of the coverage path and n_r denotes the number of operational robots. The (given) initial position of the ith robot inside $\mathcal{L}$ is represented as $\chi_i(t_0)$.

Having the above formulation in mind, the mCPP problem[2] can be transformed to calculate the robots' *paths* X_i^* $\forall i \in \{1, \ldots, n_r\}$ so as,

$$\begin{aligned} &\underset{X}{\text{minimize}}\ \max_{i\in\{1,\ldots,n_r\}} |X_i| \\ &\text{subject to } X_1 \cup X_2 \cup \cdots \cup X_{n_r} \supseteq \mathcal{L} \end{aligned} \tag{6}$$

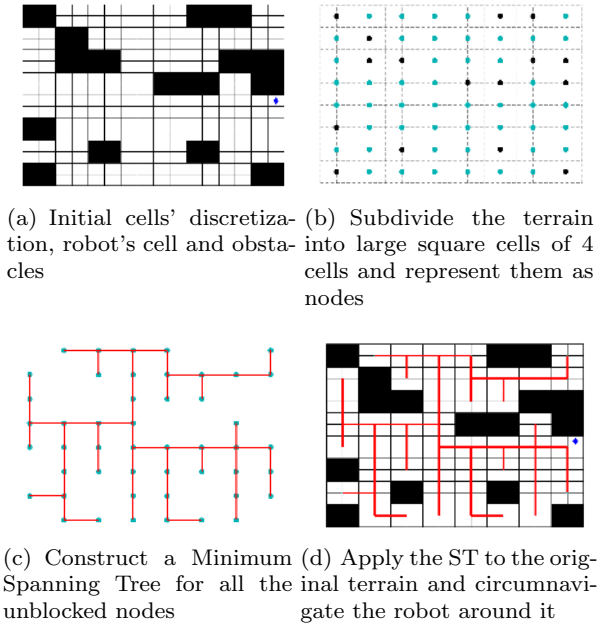
where $|X_i|$ denotes the length of the path X_i.

4 Single Robot Coverage inside Unstructured Environment

Disregarding for the moment the problem of optimal movement for a team of robots, let us consider the problem of covering a continuous unstructured area, utilizing only a single robot. Following the notation of optimization problem in equation (6), the aforementioned single robot CPP can be defined as:

$$\begin{aligned} &\underset{X_1}{\text{minimize}}\ |X_1| \\ &\text{subject to } X_1 \supseteq \mathcal{L} \end{aligned} \tag{7}$$

[2] The aforementioned formulation may include cases, where the optimal solution does not exist and therefore are neglected for the analysis. The interest reader is kindly referred to the Appendix A.



(a) Initial cells' discretization, robot's cell and obstacles

(b) Subdivide the terrain into large square cells of 4 cells and represent them as nodes

(c) Construct a Minimum Spanning Tree for all the unblocked nodes

(d) Apply the ST to the original terrain and circumnavigate the robot around it

Fig. 1 Spanning Tree Coverage Algorithm, sample execution

It has been proved in the literature that the CPP problem has an $\mathcal{O}(n)$ algorithm [18], where n is the size of grid, that is able to produce always the optimal solution. In other words, the Spanning Tree Coverage (STC) algorithm is able to construct the minimum path that coverages all the operation area $\mathcal{L}$, starting from any arbitrary unoccupied cell.

Figure 1 illustrates the basic steps of an example designing trajectory. In this approach, the terrain's cells are grouped into large square-shaped cells, each of which is either entirely blocked or entirely unblocked, and contains four of the initially discretized cells (Figure 1(b)). More precisely, the obstructed areas cannot be smaller than 4 times the size of grid's cell and this condition consists of the only algorithm's requirement. As next step, every unobstructed large cell is translated into a node (Figure 1(b)) and for every adjacent cell, an edge is introduced. For the resulting graph, a minimal spanning tree is constructed, using any minimum-spanning-tree algorithm, such as Kruskal's or Prim's algorithms [33], as it is illustrated in Figure 1(c). The robot then *circumnavigates* the spanning-tree along a (counter) clockwise direction (Figure 1(d)). The circumnavigation of the spanning tree generates a simple *closed path* X_1^*, producing an optimal -in terms of coverage time- solution.

5 Reduce the original mCPP Problem

Utilizing the findings of STC algorithm for the case of one robot, the original mCPP problem, as defined in (6), can be reduced to

$$\begin{aligned} &\underset{L}{\text{minimize}} \ \max_{i\in\{1,\dots,n_r\}} |L_i| \\ &\text{subject to } L_1 \cup L_2 \cup \cdots \cup L_{n_r} \supseteq \mathcal{L} \end{aligned} \tag{8}$$

where $L_1, L_2, \dots, L_{nr}$ denote the robot sets (and not strict paths). As next step, n_r instances of the STC algorithm could be employed -in a completely distributed manner- in order to calculate the robots' exact paths inside these sets (problem (7)). Therefore, the exploitation of STC algorithm allows the removal of the severe adjacency constraint (Definition 2) regarding the produced robot sets. In other words, only the problem of building the L_i sets, without any concern about the actual robot's movement, inside the $\mathcal{L}$ world has to be addressed.

In the rest of this section we investigate the fundamental conditions, that have to be hold regrading the L_i sets, so as the optimal solution to the overall mCPP (6) problem to be guaranteed.

Definition 3 A selection $\{L_1, L_2, \dots, L_{n_r}\}$ composes an optimal solution for the mCPP, iff

1. $L_i \cap L_j = \emptyset, \forall i, j \in 1, \dots, n_r \ , \ i \neq j$
2. $L_1 \cup L_2 \cup \cdots \cup L_{n_r} = \mathcal{L}$
3. $|L_1| \approx |L_2| \cdots \approx |L_{n_r}|$
4. L_i is connected $\forall i \in 1, \dots, n_r$
5. $\chi_i(t_0) \in L_i$

The first condition secures that every cell must be contained strictly in one robot's set, constituting the *non-backtracking* guarantee for the produced solution. The second condition states that the union of all L_i sets must contain every unblocked cell of the area to be covered (3) and depicts the fundamental coverage objective of *completeness*. The third condition establishes the *fully exploitation of the multi-robot dynamics*, by making certain that the number of cells $|L_i|$ in each robot's set are more or less the same[3]. The forth condition declares that the cells inside each robot's set L_i should be compact, forming a solid sub-region. In other words, this condition ensures that the division is *absolutely fair* and guarantees a seamlessness navigation scheme, inside spatially cohesive areas. According to that statement, no robot may spend extra/non-inclusive time to travel between unconnected areas. The final condition refines that the initial position of each robot $\chi_i(t_0)$ must be contained on its own set L_i, providing the ultimate layer of effectiveness, ensuring *zero preparation time and energy*. Any algorithm that is able to construct the L_i sets, ensuring the Definition's 3 conditions, can be utilized (in combination with the STC) to *construct optimal solutions* to the original mCPP problm (6).

Regarding to the existence of these solutions, it has been proved [27] that, a fair partition, which does not require convex pieces, always exists for any polygon and any number of partitions. The problem which is formulated here is a variation of the aforementioned one, with an extra condition, that indicates the inclusion of any arbitrary point of the polygon inside each partition. Apparently, the above problem cannot always have a solution and strongly depends on the arrangement of the arbitrary points, that need to be included in the produced fair partitions. The overall formulation of the problem along with proposed algorithm are referred to cases where at least, *one* optimal solution exists.

[3] This ambiguity is introduced mainly for two reasons. On one hand, it might be impossible to perfect divide the number of cells to be covered $|\mathcal{L}|$ with the number of robots n_r. On the other hand, even in cases where the perfect division is possible the initial configuration - placement of both the robots and obstacles - may raise limitations according to the optimal solution.

6 Divide Areas based on Robots Initial Positions (DARP)

In this section is described the DARP (Divide Areas based on Robots Initial Positions) algorithm, a specifically tailored, optimality preserving technique that divides the terrain into n_r robot-exclusive regions. To start with, DARP algorithm adopts the following cell-to-robot assignment scheme. For every ith operational robot an evaluation matrix E_i is maintained. This evaluation matrix E_i expresses the level of reachability (e.g. distance) between the cells of $\mathcal{L}$ and the ith robot's initial position $\chi_i(t_0)$. During each iteration the assignment matrix A is constructed as follows:

$$A_{x,y} = \underset{i\in\{1,\dots,n_r\}}{\text{argmin}} \ E_{i|x,y}, \ \forall (x,y) \in \mathcal{L} \tag{9}$$

Afterwards, each robot's region L_i can be computed straightforwardly by the assignment matrix A as follows:

$$L_i = \{(x,y) \in \mathcal{L} : A(x,y) = i\} \ , \ \forall i \in \{1, \dots, n_r\} \tag{10}$$

Additionally, the number of assigned cells per robot can be defined as the cardinality of the L_i set

$$k_i = |L_i| \ , \ \forall i \in \{1, \dots, n_r\} \tag{11}$$

By adopting the aforementioned cells assignments policy, regardless of the robots' evaluation matrices E, the first, second and fifth conditions of Definition 3 are always satisfied. Concretely, one cell can only be assigned to one robot (first condition), every cell has been assigned to some robot's operation plan (second condition) and it is assumed that the initial robot positions are always assigned to the corresponding robot area (fifth condition). In a nutshell, DARP algorithm is an iterative process, which appropriately modifies the robots' evaluations E_i in a *coordinated* fashion, in order to meet the two remaining -and in many cases conflicting- requirements.

Furthermore, the aforementioned cells' assignment policy automatically undertakes an additional task related to the robots' trajectories time-scheduling. If it is allowed for robots to occupy the same cells, then a fine-grained analysis should take place to prevent robot-to-robot collisions. This fact could result in a serious downgrade regarding the quality of the overall solution, even in case where the sets L_i are equal.

6.1 Equally Divide the Space

Initially, the robots evaluation matrices E_i contain distance only information:

$$E_{i|x,y} = d\left(\chi_i(t_0), [x,y]^\tau\right),\ \forall i \in \{1,\dots,n_r\} \tag{12}$$

where $d(\cdot)$ denotes the chosen distance function (e.g. Euclidean). Thus, the initial assignment matrix A (9) should be a classical Voronoi diagram.

The DARP algorithm's core idea is that each evaluation matrix E_i can be appropriately "corrected" by a term m_i as follows:

$$E_i = m_i E_i \tag{13}$$

where m_i is a scalar correction factor for the ith robot.

The third condition of Definition 3 is equivalent with the minimization of the:

$$J = \frac{1}{2}\sum_{r=1}^{n_r}\left(k_i - f\right)^2 \tag{14}$$

where f denotes the global "fair share": $f = l/n_r$ (#Unoccupied cells divided by the #robots).

A standard gradient descent method for updating m

$$m_i = m_i - \eta\frac{\partial J}{\partial m_i},\ \eta > 0,\ \forall i \in \{1,\dots,n_r\} \tag{15}$$

can be employed, in an attempt to minimize the value of the cost function (14). When attempting to apply (15), two shortcomings arise. At first, $\partial J/\partial m_i$ cannot be computed algebraically, as the analytical form that relates J with m_i is not available. On the other hand, there is no guarantee that J has only one (global) minimum.

To overcome the above problems, a cyclic Coordinate Descent (CD) methodology is adopted [35, Algorithm 1]. Coordinate descent algorithms solve optimization problems by successively performing approximate minimization along coordinate directions or coordinate hyperplanes. The global cost function is minimized cyclically over each of one of the coordinates while fixing the remaining ones at their last updated values. Each such subproblem is a scalar minimization problem, and thus can typically solved more easily than the full problem.

To start with, the global minimum of this function will always be in case where $k_1 = k_2 = \cdots = k_{n_r} = f$. Therefore, the global minimum of (14) can be obtained if we solve n_r single dimension optimization problems with the following objective function:

$$J_i = \frac{1}{2}\left(k_i - f\right)^2 \tag{16}$$

By applying the above transformation, we can achieve the following:

First and foremost, the above search is performed in local-minima free space.

Lemma 1 *All sub-problems of (16) are convex to the corresponding controllable parameter m_i.*

Proof Let's assume that the ith robot during the previous iteration, based on its evaluation matrix E_i, occupied less cells than the desirable threshold ($< f$). It is obvious from (13) and (9) that a "small" decrease in the corresponding correction factor, m_i (< 1), will lead to an increase in the number of assigned cells k_i, assuming that the other robots' evaluation matrices E remain the same. Therefore, the corresponding objective function J_i (16) will be decreased. Although, if we "over-decrease" the m_i factor "many" cells will be assigned to the ith robot. Now, the J_i will start to rise again, as the k_i will be greater that the f. From this point and after, if we continue to decrease m_i, the ith robot will be assigned to more and more cells, as k_i can only be increased in response to m_i decrease. The value of J_i is saturated when all the available cells[4] ($l - n_r + 1$) have been assigned to the ith cell, and further decreases of m_i cannot affect neither k_i nor J_i. Hence, J_i will monotonically be increased, as m_i is decreased until the maximum possible $J_i\,|_{k_i = l - n_r + 1} = \frac{1}{2}\left(\frac{(l-n_r)(n_r-1)}{n_r}\right)^2$.

[4] The available cells are $l - n_r + 1$ as the initial robot cells are a-priori allocated to the corresponding robot.

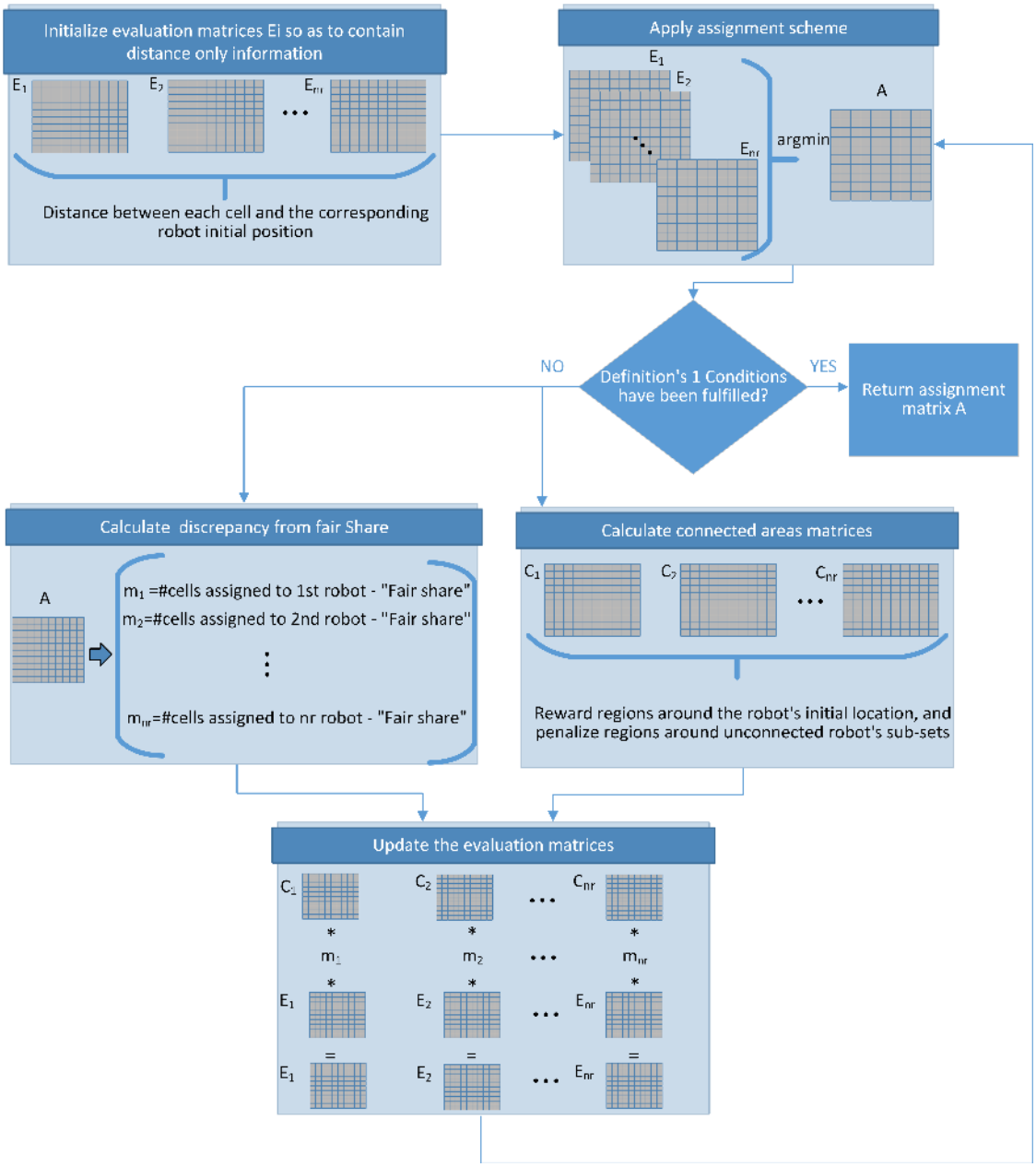


Fig. 2 DARP algorithm flowchart - Divide Areas based on Robots Initial Positions

Therefore, the previously encountered minimum is the global one. The proof continues to hold if, instead, we had assumed that the ith robot had been assigned to more cells than f. □

Additionally, the update rule of m_i can be straightforwardly calculated for each objective function (16) separately as:

$$\begin{aligned} m_i &= m_i - \eta \frac{\partial J_i}{\partial m_i} \\ &= m_i - \eta \left(k_i - f\right) \frac{\partial k_i}{\partial m_i} \end{aligned} \tag{17}$$

Due to the nature of the problem, the changes in k_i with respect to m_i will always be negative (see proof in Lemma 1) and they are almost identical for each robot (for a given sub-problem (16)). Additionally, two sets of evaluation matrices $\{E_1, \ldots, E_{n_r}\}$ and $\{\alpha E_1, \ldots, \alpha E_{n_r}\}$, where α denotes any positive constant, correspond to the identical assignment matrices (9). Therefore, the influence of $|\partial k_i / \partial m_i|$ can be securely omitted and the final update policy can be approximated as follows:

$$m_i = m_i + c\left(k_i - f\right) \tag{18}$$

where c denotes a positive tunable parameter.

Summarizing, using Lemma 1 we can establish that even thought the global cost function f can be generally non-convex - depending of the robots and obstacles formation - with many local minima , each robot's contribution J_i is a convex function with respect to the controllable parameter m_i. As shown in [37], cyclic Coordinate Descent methodologies, where the above property holds, are able to converge to a global optimal solution set m^*, i.e.

$$J(m^*) \leq J(m),\ \forall m \in \mathrm{dom}(J) \tag{19}$$

with respect to the initial evaluation matrices E_i (12).

6.2 Build Spatial Connected Areas

Although, the aforementioned procedure can be easily converge to share the available cells $\mathcal{L}$ among the different robots, it cannot guarantee the continuity of each robot's sub-region (condition 4, Definition 3). In oder to deal with such situations, for every ith robot that occupies more than one distinct regions the following matrix is introduced

$$\mathcal{C}_{i|x,y} = \min\left(||[x,y]-r||\right) - \min\left(||[x,y]-q||\right), \quad \forall r \in \mathcal{R}_i,\ q \in \mathcal{Q}_i \tag{20}$$

where $\mathcal{R}_i$ denotes the connected set of cells where the ith robot actual lies in ($\chi_i(t_0)$) and $\mathcal{Q}_i$ denotes the union of all other connected sets, which have been assigned to the ith robot but they do not have spatial connectivity with $\mathcal{R}_i$ set. In a more abstract conceptualization, the $\mathcal{C}_i$ is constructed in a way, to reward the regions around the ith robot location's subset, and to penalize the regions around other unconnected subsets, constructing gradually a closed-shape region. If all the assigned cells to ith robot belong to the same closed-shape region, the $\mathcal{C}_i$ is set to be the all-one-matrix.

The final update in the ith evaluation matrices is calculated as

$$E_i = \mathcal{C}_i \odot (m_i E_i) \tag{21}$$

where $\odot$ denotes the element-wise multiplication. The findings of the previous subsections are illustrated in figure 2, where a flowchart of the proposed algorithm is presented.

6.3 Performance Discussion

Although simple in concept, the DARP algorithm aims to provide the optimal cells' assignment, in cases where at least one exists. A sample execution is illustrated in Figure 3, where the terrain is constituted of 42×42 cells and the number of robots is $n_r = 5$. The initial robots' positions were squeezed inside a sub-region of the whole operation area, at the left bottom space of the grid, with dimension 10×10 cells. Each sub-figure illustrates the condition of the assignment matrix A (9) at the corresponding iteration. Apparently, the algorithm was terminated after 260 iterations, fulfilling all the conditions of Definition 3[5].

It is worth highlighting, that contrary to robot's evaluation matrix E_i which is continuous, the produced sub-areas that are finally assigned to each robot, may be arbitrary unconnected (at least temporary, e.g. figure 3(b)) non-convex areas. In fact, this DARP algorithm's key feature, allows the gradually inclusion to each robot's sub-region, of any arbitrary located cell. More precisely, DARP algorithm is capable of escaping the local minima by *temporarily violating the condition about the connectivity of the each ith robot assignment matrix*. Afterwards, the algorithm gradually eliminates the presence of unconnected areas, by reinforcing the robot's evaluation E_i around the original (the one that the robot actually lies in) sub-area. By the time, the connectivity inside the exclusive robot sets L_i is restored, the evaluation matrices E_i will have completely changed their forms, and ideally towards to the optimal cells assignment.

The proposed algorithm diverges from the general class of *local search* algorithms in the sense that, it changes its current state, mainly based on the global optimal one and not only by evaluating information from the current and the candidate states. Over and above, DARP algorithm approximates the behavior of a gradient decent algorithm, with an extra capability to search effectively and reach the global optimal, even in case with multiple local minima.

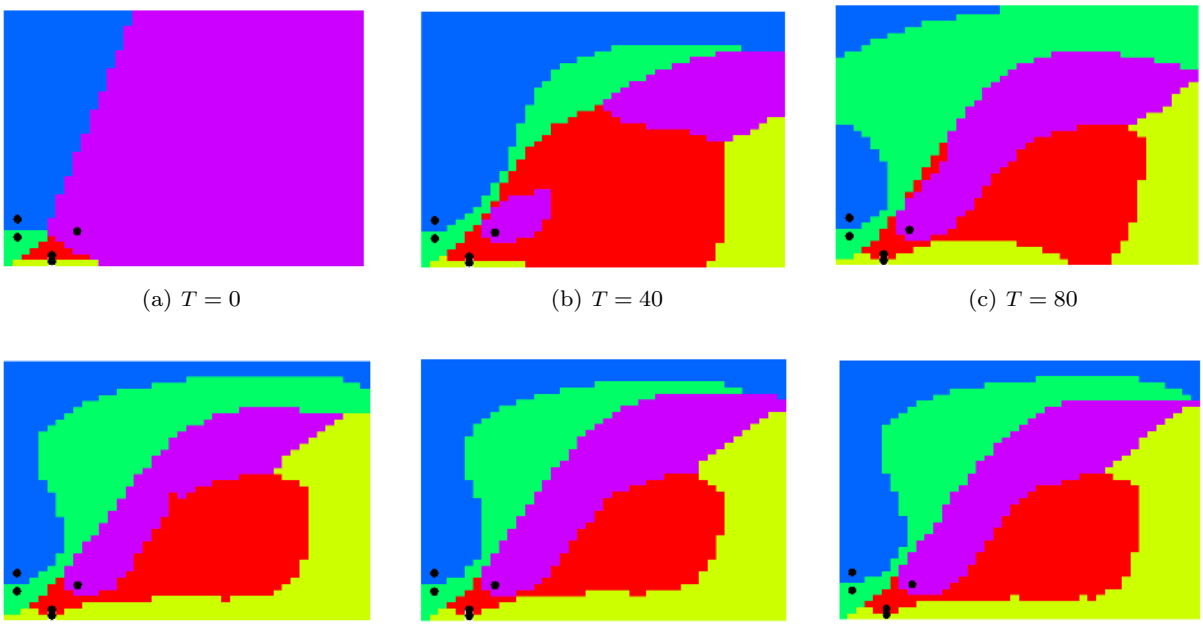
6.4 Computational & Memory Complexity Analysis from an Approximation Point of View

The memory needs of the algorithm can be calculated straightforwardly, as it utilizes a constant number β of matrices with dimensions $(n_r \times n)$. In other words, the algorithm's memory complexity is linear to the size of input $(n_r \times n)$, i.e. $\mathcal{O}(\beta \times n_r \times n)$.

The main optimization loop performs $\alpha \times n_r \times n$ operations, where α is a constant number, resulting in $\mathcal{O}(\alpha \times n_r \times n)$[6] computational complexity. However,

[5] The interested readers are kindly referred to http://tinyurl.com/DARP-live to watch an additional recorded execution of the DARP algorithm.

[6] Please note that, in both complexity calculations, there is an additive constant which is omitted, due to its negligible influence



(a) $T = 0$ (b) $T = 40$ (c) $T = 80$

(d) $T = 120$ (e) $T = 200$ (f) $T = 260$

Fig. 3 Progression of the robots sub-regions over iterations

the number of times which the main optimization loop is executed (MaxIter) is not constant or linear, but it depends on the specific characteristics of the current problem in a non-linear fashion. As it is not possible to find the closed form that relates the maximum needed (main optimization loop) iterations with the number of robots n_r, initial deployments $\chi_i(t_0)$ and the grid size n, the following approximation scheme for the algorithm's computational needs is adopted.

A series of simulations were conducted in order to measure the MaxIter (main optimization loop) iterations that were needed for the construction of the optimal solution (Definition 3). For each configuration (n_r and n) the results were validated by repeating the experiment with different, randomly chosen, initial $\chi_i(t_0)$, in order to be able to approximate the MaxIter for the worst case scenario.

Please note that, it is practically infeasible to compute exhaustively, the actual worst case for each configuration, due to the vast number of possible combinations of the initial robots positions. Nonetheless, in every different set-up the number of randomly created instances was proportional to the input parameters (n_r and n). By doing so, it is ensured that the computed worst case complexity is representative of the number of possible occurring configurations. The number of the experiments for each configuration starts form 50 for $\{n_r = 3, n = 500\}$ and reaches up to 5000 for $\{n_r = 20, n = 5000\}$, constructing a pool of more than 120000 different experiments.

In order to present the overall approximation on the DARP's complexity (MaxIter $\times\, n_r \times n$), for each $\{n_r, n\}$ scenario was extracted the worse-case (maximum) of the needed iterations (MaxIter). These worse cases for each scenario are translated into a surface by applying a polynomial least squares curve fitting technique. The produced surface is illustrated with blue color in figure 4, where the operations' needs growth, with respect to the input, is representing both in linear and logarithmic scale. Moreover, and in order to evaluate the produced complexity results a number of polynomial surfaces is utilized. More specifically, with yellow, magenta and green color is illustrated the complexity curves in cases of $f_1(n_r, n) = n_r^2 \times n^2$, $f_2(n_r, n) = n_r^3 \times n^2$, and $f_3(n_r, n) = n_r^2 \times n^3$, respectively. The evidences of this representation indicate that DARP's complexity is cubic with respect to the input of the problem ($n_r \times n$), as the approximation on the complexity curve is strictly bounded under the $n_r^3 \times n^2$ curve, at least until the maximum simulated parameters $n_r = 20$ and $n = 5000$.

Concluding this section, it is worth mentioned that the proposed algorithm cannot bypass the NP-nature of the mCPP problem, but it provides an approximately polynomial algorithm until a specific (practical interesting) input. If both the size of the robots and number of the cells grow beyond the aforementioned order

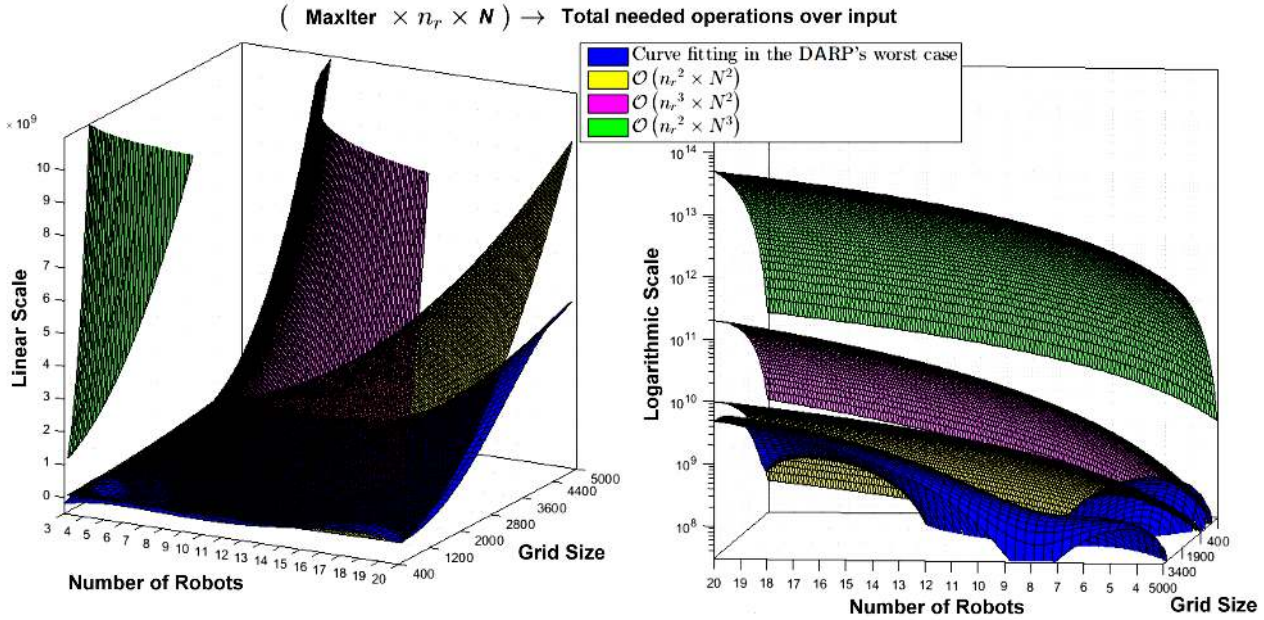


Fig. 4 Approximation on DARP's complexity, a comparison with known polynomial surfaces

of magnitude of the input, the algorithm may lose its polynomial behavior.

6.5 Beyond the classical mCPP

It is worth to point out that, the DARP algorithm is an optimization based one, which allows the inclusion of other secondary objectives, depending on the final multi-robot application, such as robot's subareas smoothing etc., by just revising the appropriate performances' criteria. In the literature, the problem of mCPP is usually defined as in section 3, where it is desirable to produce balanced paths, in order to exploit all the available robots' capabilities. However, there might be cases where specific robots' characteristics (e.g. sensing module, battery life, etc.) impose different utilization portions among the different robots. The proposed approach is able to straightforwardly encompass this additional information, by appropriately modifying the calculation of J_i (16). More precisely, the objective function for the ith robot is going to alternate as

$$J_i = \frac{1}{2}\left(k_i - p_i\right)^2 \tag{22}$$

where p_i is the corresponding portion of the map that the ith robot has to covered based on its capabilities or limitations ($\sum_{i=1}^{n_r} p_i = 1$).

However in order to be in-line with the ordinary mCPP formulation we limit our simulation evaluation (section 8) only to scenarios where an equal cells' division between the robots is considered desirable.

7 Overview of the proposed multi-robot coverage path planning algorithm

This section summarizes the complete algorithm for mCPP problem (6), by fusing the findings of the DARP and STC algorithms. The proposed algorithm is separated into two phases: During the first phase, the DARP algorithm divides the cells of $\mathcal{L}$ set into n_r exclusive areas L_i, for each available robot, as explained in section 6. The outcome L_i of that process serves as the operational area for each robot separately (section 4).

After the applying of DARP algorithm and the corresponding production of L_i sets, the original multi robot optimization problem (6) is downgraded to n_r single robots CPP problems, alleviating its explosive combinatorial complexity. Each one of these problems can be expressed as:

$$\begin{aligned} &\underset{X_i}{\text{minimize}} \ \ |X_i| \\ &\text{subject to } X_i \supseteq L_i \end{aligned} \tag{23}$$

where X_i denotes a robot path as defined in Definition 2. As shown in section 4 this class of optimization problems (single robot inside grid connected environments) can be solved in an optimum manner (*optimal solution - polynomial time*), utilizing the STC Algorithm.

Even though the final path $\{X_1, X_2, \ldots, X_{n_r}\}$ construction takes place in a fully distributed manner, the union of the produced solutions is actually an optimal solution for the eq. (6) problem, without any compromise in the quality or the generality of the solution. In

essence, this can be attained by the original construction, during the first phase (section 6), of the L_i sets, ensuring that the conditions of the Definition 3 are satisfied. The aforementioned feature of the algorithm not only allows the fully parallelization of the algorithm, but dramatically reduces the complexity of the initial mCPP problem to the order of magnitude of the STC algorithm.

Figure 5 depicts an example execution of the proposed algorithm. Sub-figure 5(a) illustrates the initial robots positions along with the placements of the fixed obstacles. The sub-figure 5(b) represents the result from the area division approach, as described in figure 2. Each sub-area's Minimum Spanning Tree is represented in sub-figure 5(c) with spatial information about the nodes inside the $\mathcal{L}$ world. Finally, the proposed algorithm let the robots move along the path that circumnavigates the corresponding spanning tree, as is shown in sub-figure 5(d). It is worth noticing that, the produced paths constitute an optimal solution, as the number of cells that have been assigned to each robots are [12 13 12 12 12 13 12 12 12] (Definition 3, condition 3)). The corresponding summation, translated to the operational world, is $4\,(12 * 7 + 13 * 2) = 440$, which is exactly the number of cells to be covered (Definition 3, condition 2)).

8 Simulation Results

This section presents a comparison study between the proposed DARP+STC algorithm and two of the state-of-the-art methods ("MFS" and "Optimized MSTC" see related work). In order to produce comparable results, we adopt the same simulation set-up as in [39]. More precisely:

- The size of the terrain is always $[rows, cols] = 98 \times 98$.
- We considered two kind of terrains: 1) The empty terrain [empty] and 2) the one, which has the 10% of its cells occupied by obstacles [outdoor]. The obstacles' arrangement follows a random uniform distribution.
- The number of robots varies from 2, 8, 14 to 20 robots.
- The robots initial placement can take three different types, according to their in-between maximum distance (clustering). More precisely, the maximum distance between two robots can be at most 1) 30% [30] or, 2) 60% [60] of the maximum terrain's dimension correspondingly, and 3) without any distance constraint (free selection) [none].

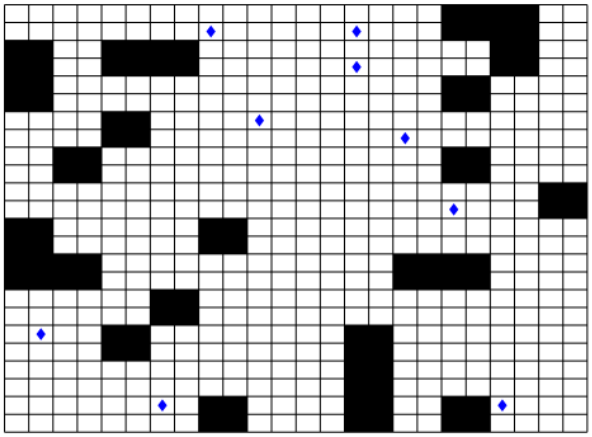
In order to obtain a fair comparison with MFC and Optimized MSTC algorithm, we repeated each scenario 100 times. The results for each combination of different evaluation scenario and algorithm, are illustrated in Table 8, where it is reported the maximum [Max] and minimum [Min] coverage time for all robots, in terms of paths lengths. Simultaneously, for each scenario we provide the idealized coverage time [Ideal Max], which represents the optimal solution to the problem. In other words, this value is simply calculated by dividing the number of unoccupied cells with the number of robots (f). Apparently, the larger deviations from the ideal coverage time, the bigger the difference between the robots paths, resulting in unbalanced, sub-optimal routes. The overall scoring for each scenario per algorithm, against the ideal coverage time, is depicted in [Ratio] column and reports the ratio of actual (maximum) traveled path and the ideal coverage time.

The direct observation is that the performance of the proposed algorithm DARP+STC seems to be immune to the number of robots and/or the obstacles and/or the initial clustering of the robots, as it performs with almost the same ratio over the different scenarios. Additionally, all the results are close to the [Ideal Max], and the maximum difference between two robots path is at most 4 cells, independent of the number of robots or/and the grid size, i.e. $\| |X_i| - |X_j| \| \leq 4,\ \forall i, j \in 1, \ldots, n_r$. The above effectiveness bound is straightforwardly incoming from the DARP algorithm optimality guarantee. The DARP algorithm calculates the L_i areas, having at most 1 cell difference among the different ith robots (see figure 2). This maximum discrepancy is translated into 4 cells after the appliance of STC algorithm (section 4). Overall, these findings seal experimentally, the performance of the proposed algorithm.

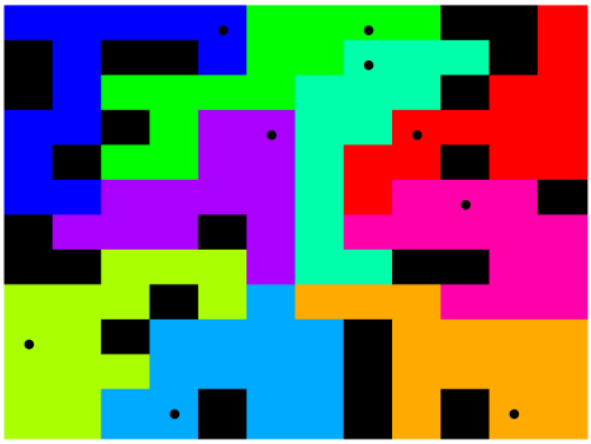
The afore-mentioned optimal performance does not come without shortcomings. In all cases, initial configurations that lead to sub-optimal results are discarded from the pool of test cases, while both the other two algorithms are able to straightforwardly produce some sub-optimal operation plans. A proper categorization of the cases where optimal solutions cannot be obtained, is provided in A, where also preliminary solutions, in-line with the proposed approach, are also presented.

9 Conclusions and Future Work

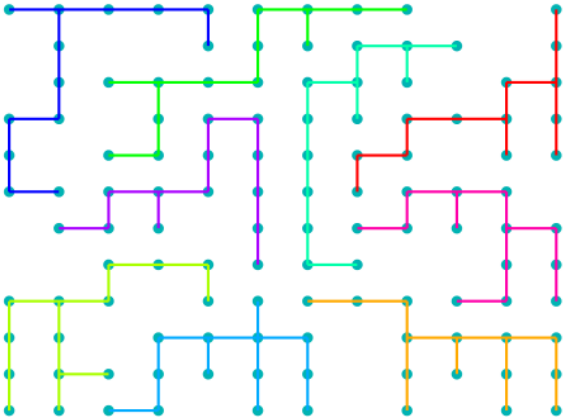
The proposed approach orchestrates the optimal coordination of a multi-robot team, so as to completely cover an area of interest. During the preliminary analysis, the underlying mCPP problem is translated into a constraint satisfaction problem, by formally define the exact attributes that have to be hold in order to achieve



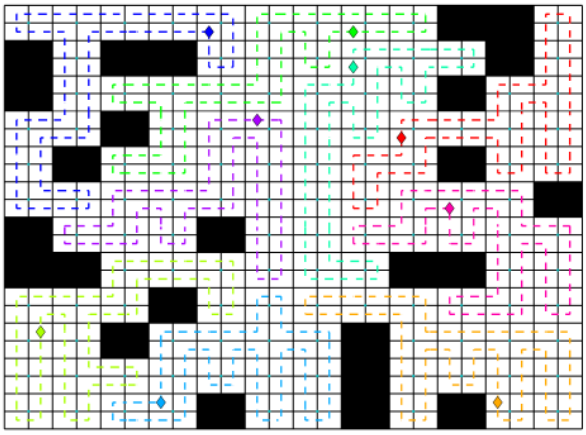
(a) Initial cells discretization, robots cell and obstacles



(b) DARP outcome - robots' exclusive areas



(c) Constructing Minimum Spanning Trees for each one of the robots sets



(d) Final Paths, designed to circumnavigate the MSTs

Fig. 5 DARP+STC Proposed Approach, sample execution with 24x24 grid size, 9 robots and 100 obstacles

the optimal performance. In heart of the proposed approach lies the DARP methodology, a search algorithm, which finds the optimal cells assignment for each robot utilizing a cyclic coordinate descent approach, which takes into account both the robots initial positions and the obstacles formation. The outcome of the DARP algorithm constitutes a set of exclusive operation areas for each mobile robot. These well-defined regions, are forwarded to each robot's planner, where by employing STC algorithm, the exact route that covers the assigned area is calculated. The overall navigation scheme achieves to traverse the *complete* operation area, without *backtracking* in already visited areas, *starting from the exact initial* robot positions. To the best of our knowledge, no other method from the literature exhibits all the aforementioned features at the same time.

Several avenues of exploration are left open for future work. One direction could be the relaxing of one or more constraints of Definition 3. For instance, in expense of the non-backtracking attribute, the produced paths can be constructed to be convex only (less messy) or/and the shape of the STC can be appropriately modified in order of the turns in robots' paths to be minimized. In addition, we intend to include in our methodology another stage, which will be in charge for the automatic recognition/detection of non-optimal cases, in order to directly apply the appropriate, predefined solution scheme. Finally, in our future plans is the development of an online version of DARP algorithm, so as to be able to operate inside *completely unknown* terrains.

Terrain	Robots	Clustering	Ideal Max	DARP+STC			MFC			Optimized MSTC		
				Max	(Min)	Ratio	Max	(Min)	Ratio	Max	(Min)	Ratio
Empty	2	30	4801	4803	(4799)	1.001	4878	(4731)	1.02	5337	(4410)	1.11
	2	60	4801	4803	(4799)	1.001	4886	(4720)	1.02	5513	(4241)	1.15
	2	none	4801	4803	(4799)	1.001	4888	(4725)	1.02	5602	(4168)	1.17
	8	30	1200	1203	(1199)	1.003	1399	(838)	1.17	3817	(45)	3.18
	8	60	1200	1203	(1199)	1.003	1415	(904)	1.18	3539	(93)	2.95
	8	none	1200	1203	(1199)	1.003	1394	(956)	1.16	3281	(146)	2.73
	14	30	685	687	(683)	1.006	841	(431)	1.23	3756	(5)	5.48
	14	60	685	687	(683)	1.006	819	(522)	1.20	3461	(16)	5.05
	14	none	685	687	(683)	1.006	830	(513)	1.21	3072	(40)	4.48
	20	30	479	483	(479)	1.008	615	(307)	1.28	3685	(3)	7.69
	20	60	479	483	(479)	1.008	604	(332)	1.26	3439	(9)	7.18
	20	none	479	483	(479)	1.008	604	(321)	1.26	2867	(18)	5.99
Outdoor	2	30	4321	4321	(4321)	1	4380	(4269)	1.01	4772	(4031)	1.10
	2	60	4321	4321	(4321)	1	4382	(4266)	1.01	4854	(3954)	1.12
	2	none	4321	4321	(4321)	1	4377	(4269)	1.01	4923	(3903)	1.14
	8	30	1079	1082	(1078)	1.003	1263	(789)	1.17	3561	(26)	3.30
	8	60	1079	1082	(1078)	1.003	1278	(790)	1.18	3229	(70)	2.99
	8	none	1079	1082	(1078)	1.003	1247	(873)	1.16	3099	(94)	2.87
	14	30	616	620	(616)	1.006	746	(450)	1.24	3452	(6)	5.60
	14	60	616	620	(616)	1.006	750	(482)	1.22	3228	(20)	5.24
	14	none	616	620	(616)	1.006	746	(464)	1.21	2819	(37)	4.58
	20	30	431	434	(430)	1.007	572	(280)	1.33	3437	(3)	7.97
	20	60	431	434	(430)	1.007	557	(285)	1.29	3140	(9)	7.29
	20	none	431	434	(430)	1.007	551	(296)	1.28	2740	(18)	6.36

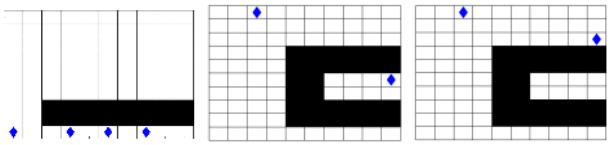
Table 1 Cover time (in terms of path length) for DARP+STC, compared with MFC and Optimized MSTC

A Cases where the optimal solution does not exist

The problem formulation, as it is defined in section 3, it may contain cases where the given placement of the obstacles or the robots blocks the access to one or more cells. Although these cases are considered out of the scope of the paper, and excluded from the considered scenarios, here in the appendix we categorize them and propose some preliminary solutions in-line with the proposed approach.

The first class consists of cases where an optimal solution to the mCPP problem can not be attained, due to the initial placements of the robots (sub-figure 6(a)). In these cases, one could spend some preparatory steps in order to rearrange the robots, so as to transform the problem into a solvable scenario (by the proposed approach DARP+STC). This rearrangement is not trivial and is forming another optimization problem, where now the objective is to find the minimum path to travel in order to render the problem tractable. Alternatively, one could apply a relaxed version of DARP algorithm by removing its non-backtracking property (Definition 3, condition 1).

Another case, where the coverage task cannot be equally separated among the available robots, might be occurred, where one or more robots are trapped inside non-avoided, bounded sub-areas (sub-figure 6(b)). In these cases one could straightforwardly apply the proposed approach, as many times as the number of bounded zones, and the optimal attainable solutions is again guaranteed. Apparently, in this case it is highly unlikely to end up having a balanced path length across all the robots' planners. In fact, now the produced path lengths are highly dependent on the size of the corresponding bounded area. However, different robots that lie in same sub-area should have almost the same workload (Definition 3, condition 3).



(a) The robots initial placement limits some robots operation plans

(b) The obstacles and robots placement forms two exclusive sub-areas

(c) The obstacles do not allow the fully coverage of the area of interest

Fig. 6 Cases where the robots and/or obstacles arrangement, do not allow the acquisition of optimal solution

Moreover, there are non-recoverable cases, where one or more sub-areas cannot be reached (sub-figure 6(c)). In such situations the proposed algorithm can be applied on the remaining terrain, ensuring the optimal robots' path construction. Finally, it might be occurred a combination of the above scenarios and then one could apply a hybrid version of the aforementioned solutions.

Over and above, it should be highlighted that, in all these cases the fact that the proposed approach is not able to deliver an optimal set of paths, is not some kind of weakness, but it is due to the fact that the optimal solution, at least with the properties as defined in Definition 3, does not exist.

Acknowledgments

This project is funded by the European Commission (FIRE+ challenge, Horizon 2020) that aims to provide for research, technological development and demonstration under grant agreement no 645220 (RAWFIE)

References

1. irobot web site. `http://www.irobot.com`. Accessed: 2016-4-1.
2. Mixed integer linear programming (milp) solver, software. `http://lpsolve.sourceforge.net/`. Accessed: 2016-4-1.
3. The area partitioning problem. In *Computational Geometry, Fredericton, New Brunswick, Canada, August 16-19, 2000. Proceedings of the12th Canadian Conference on*, 2000.
4. Ercan Acar, Yangang Zhang, Howie Choset, Mark Schervish, Albert G Costa, Renata Melamud, David C Lean, and Amy Graveline. Path planning for robotic demining and development of a test platform. In *International Conference on Field and Service Robotics*, volume 1, pages 161–168, 2001.
5. Noa Agmon, Noam Hazon, Gal Kaminka, et al. Constructing spanning trees for efficient multi-robot coverage. In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pages 1698–1703. IEEE, 2006.
6. Dimitrios S Apostolopoulos, Liam Pedersen, Benjamin N Shamah, Kimberly Shillcutt, Michael D Wagner, and William L Whittaker. Robotic antarctic meteorite search: Outcomes. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 4, pages 4174–4179. IEEE, 2001.
7. Franz Aurenhammer. Voronoi diagramsa survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991.
8. Antonio Barrientos, Julian Colorado, Jaime del Cerro, Alexander Martinez, Claudio Rossi, David Sanz, and João Valente. Aerial remote sensing in agriculture: A practical approach to area coverage and path planning for fleets of mini aerial robots. *Journal of Field Robotics*, 28(5):667–689, 2011.
9. Andreas Breitenmoser, Mac Schwager, Jean-Claude Metzger, Roland Siegwart, and Daniela Rus. Voronoi coverage of non-convex environments with a group of networked robots. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 4982–4989. IEEE, 2010.
10. Zack J Butler, Alfred A Rizzi, and Ralph L Hollis. Contact sensor-based coverage of rectilinear environments. In *Intelligent Control/Intelligent Systems and Semiotics, 1999. Proceedings of the 1999 IEEE International Symposium on*, pages 266–271. IEEE, 1999.
11. Howie Choset. Coverage for robotics–a survey of recent results. *Annals of mathematics and artificial intelligence*, 31(1-4):113–126, 2001.
12. Jorge Cortés. Coverage optimization and spatial load balancing by robotic sensor networks. *Automatic Control, IEEE Transactions on*, 55(3):749–754, 2010.
13. Jorge Cortes, Sonia Martinez, Timur Karatas, and Francesco Bullo. Coverage control for mobile sensing networks. In *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, volume 2, pages 1327–1332. IEEE, 2002.
14. M Bernardine Dias, Robert Zlot, Nidhi Kalra, and Anthony Stentz. Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, 94(7):1257–1270, 2006.
15. Qiang Du, Maria Emelianenko, and Lili Ju. Convergence of the lloyd algorithm for computing centroidal voronoi tessellations. *SIAM journal on numerical analysis*, 44(1):102–119, 2006.
16. Joseph W Durham, Ruggero Carli, Paolo Frasca, and Francesco Bullo. Discrete partitioning and coverage control for gossiping robots. *IEEE Transactions on Robotics*, 28(2):364–378, 2012.
17. Yehuda Elmaliach, Noa Agmon, and Gal A Kaminka. Multi-robot area patrol under frequency constraints. *Annals of Mathematics and Artificial Intelligence*, 57(3-4):293–320, 2009.
18. Yoav Gabriely and Elon Rimon. Spanning-tree based coverage of continuous areas by a mobile robot. *Annals of Mathematics and Artificial Intelligence*, 31(1-4):77–98, 2001.
19. Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12):1258–1276, 2013.
20. Ken Goldberg. Robotics: Countering singularity sensationalism. *Nature*, 526(7573):320–321, 2015.
21. Noam Hazon, Gal Kaminka, et al. Redundancy, efficiency and robustness in multi-robot coverage. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 735–741. IEEE, 2005.
22. Athanasios Kapoutsis, SA Chatzichristofis, Lefteris Doitsidis, Joao Borges de Sousa, Elias B Kosmatopoulos, et al. Autonomous navigation of teams of unmanned aerial or underwater vehicles for exploration of unknown static & dynamic environments. In *Control & Automation (MED), 2013 21st Mediterranean Conference on*, pages 1181–1188. IEEE, 2013.
23. Athanasios Ch. Kapoutsis, Savvas A. Chatzichristofis, Lefteris Doitsidis, João Borges de Sousa, Jose Pinto, Jose Braga, and Elias B. Kosmatopoulos. Real-time adaptive multi-robot exploration with application to underwater map construction. *Autonomous Robots*, 40(6):987–1015, 2016.
24. Stuart P Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982.
25. Ivan Maza and Anibal Ollero. Multiple uav cooperative searching operation using polygon area decomposition and efficient coverage algorithms. In *Distributed Autonomous Robotic Systems 6*, pages 221–230. Springer, 2007.
26. David Moloney and Oscar Deniz Suarez. A vision for the future [soapbox]. *Consumer Electronics Magazine, IEEE*, 4(2):40–45, 2015.
27. R Nandakumar and N Ramana Rao. Fair partitions of polygons: An elementary introduction. *Proceedings-Mathematical Sciences*, 122(3):459–467, 2012.
28. Mark Ollis and Anthony Stentz. Vision-based perception for an automated harvester. In *Intelligent Robots and Systems, 1997. IROS'97., Proceedings of the 1997 IEEE/RSJ International Conference on*, volume 3, pages 1838–1844. IEEE, 1997.
29. Domènec Puig, Miguel Angel García, and L Wu. A new global optimization strategy for coordinated multi-robot exploration: Development and comparative evaluation. *Robotics and Autonomous Systems*, 59(9):635–653, 2011.
30. Ariel Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica: Journal of the Econometric Society*, pages 97–109, 1982.
31. D Scaramuzza, MC Achtelik, L Doitsidis, F Fraundorfer, EB Kosmatopoulos, A Martinelli, MW Achtelik, M Chli, SA Chatzichristofis, L Kneip, et al. Vision-controlled micro flying robots: from system design to autonomous navigation and mapping in gps-denied environments. *IEEE Robotics & Automation Magazine*, pages 1–10, 2014.

32. Mac Schwager, Daniela Rus, and Jean-Jacques Slotine. Decentralized, adaptive coverage control for networked robots. *The International Journal of Robotics Research*, 28(3):357–375, 2009.
33. Robert Endre Tarjan. *Data structures and network algorithms*, volume 14. SIAM, 1983.
34. Sonia Waharte and Niki Trigoni. Supporting search and rescue operations with uavs. In *Emerging Security Technologies (EST), 2010 International Conference on*, pages 142–147. IEEE, 2010.
35. Stephen J Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1):3–34, 2015.
36. Anqi Xu, Chatavut Viriyasuthee, and Ioannis Rekleitis. Efficient complete coverage of a known arbitrary environment with applications to aerial operations. *Autonomous Robots*, 36(4):365–381, 2014.
37. Yangyang Xu and Wotao Yin. A block coordinate descent method for regularized multiconvex optimization with applications to nonnegative tensor factorization and completion. *SIAM Journal on imaging sciences*, 6(3):1758–1789, 2013.
38. Zhiyang Yao. Finding efficient robot path for the complete coverage of a known space. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3369–3374. IEEE, 2006.
39. Xiaoming Zheng, Sonal Jain, Sven Koenig, and David Kempe. Multi-robot forest coverage. In *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 3852–3857. IEEE, 2005.