Democritus University of Thrace
School of Engineering
Department of Electrical & Computer Engineering
Division of Electronics and Information Systems Technology
Automatic Control Systems and Robotics Laboratory

Towards a fully autonomous and cooperative deployment of multi-robot teams for exploration and coverage in unknown or partially known environments

Doctoral Dissertation of:

Athanasios Kapoutsis

Advisor:
Prof. Elias Kosmatopoulos

Xanthi, December 2017

ΔΗΜΟΚΡΙΤΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΡΑΚΗΣ
ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ ΞΑΝΘΗΣ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΕΡΓΑΣΤΗΡΙΟ ΑΥΤΟΜΑΤΟΥ ΕΛΕΓΧΟΥ ΚΑΙ ΡΟΜΠΟΤΙΚΗΣ

ΠΛΗΡΩΣ ΑΥΤΟΝΟΜΕΣ ΚΑΙ ΣΥΝΕΡΓΑΤΙΚΕΣ ΜΕΘΟΔΟΙ ΓΙΑ ΟΜΑΔΕΣ ΡΟΜΠΟΤ ΜΕ ΣΤΟΧΟ ΤΗΝ ΕΞΕΡΕΥΝΗΣΗ ΚΑΙ ΚΑΛΥΨΗ ΑΓΝΩΣΤΩΝ Ή ΜΕΡΙΚΩΣ ΓΝΩΣΤΩΝ ΠΕΡΙΟΧΩΝ

ΔΙΔΑΚΤΟΡΙΚΗ ΔΙΑΤΡΙΒΗ του

Αθανασιος Καποутсης

Επιβλέπων:
Η. Κοσματοπουλος, Καθ. ΔΠΘ

Ξάνθη, Δεκέμβριος 2017

Η έγκριση της διδακτορικής διατριβής από το Τμήμα Ηλεκτρολόγων Μηχανικών της Πολυτεχνικής Σχολής του Δημοκρίτειου Πανεπιστημίου Θράκης, δεν υποδηλώνει την αποδοχή οποιασδήποτε γνώμης του συγγραφέα.

Η ΕΠΤΑΜΕΛΗΣ ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ:

Ο Επιβλεπων
Η. Κοσματοπουλος, Καθ. ΔΠΘ

Τα Μελη
Ι. Μπουταλης, Καθ. ΔΠΘ

Γ. Ροβιθακης, Καθ. ΑΠΘ

Σ. Ρουμελιωτης, Καθ. Παν/μιο Μινεσοτας

Α. Γαστερατος, Καθ. ΔΠΘ

Σ. Χατζηχριστοφης, Αν. Καθ. Παν/μιο Νεαπολις Παφος

Μ. Λαγουδακης, Αν. Καθ. Πολ/χνειο Κρητης

Acknowledgements

First of all, I would like to thank my advisor Professor Elia Kosmatopoulo for his continuous support on my Phd studies and related research. His research capabilities together with his genuine modesty were an inspiration to me. I consider myself extremely lucky to have done my Phd under his supervision.

A debt of gratitude goes to my friend Savva Chatzichristofi for our long-lasting and productive collaboration, for his assistance in my research efforts and his priceless contribution in my published (and under-review) papers.

I also need to thank all my friends that helped me to have an as-normal-as-it-can-be-for-a-Phd-student lifestyle, without whom I would be completely lost.

It would be a serious omission not to thank my brother Pavlo, for all the stimulating discussions and all the fun we have had, mainly while solving hackerrank problems.

I would like to sincerely thank Christina Malliou for standing patiently by my side all these years, always willing to sacrifice a lot of her time for providing well-thought-out advices and support; I would not have made it to this point without her.

Above all, I thank my parents for their unconditioned love, support, and faith in me, not only during the last five years but during the last twenty-eight years. I dedicate this thesis to them.

Xanthi,
December 2017 *Thanasis*

The sad thing about artificial intelligence is that it lacks artifice and therefore intelligence

Baudrillard

Abstract

In this thesis we deal with the problem of navigating a team of robots in both known and unknown environments, so as the mission's objectives to be fulfilled. The structure of this thesis is divided into two main pillars. In the first pillar we deal with the problem of determining an optimal path involving all points of a given area of interest (offline), while avoiding sub-areas with specific characteristics (e.g. obstacles, no-fly zones, etc.). This problem, which is usually referred as multi-robot coverage path planning (mCPP), has been proven to be NP-hard. Currently, existing approaches produce polynomial algorithms that are able to only approximate the minimum covering time. In chapter 3, a novel methodology is proposed, capable of producing such optimal paths in approximately polynomial time. 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, in such a way to guarantee: complete coverage, non-backtracking solution, minimum coverage path, while at the same time does not need any preparatory stage. In the second pillar of this thesis, we design algorithms capable of navigating team of robots without any prior knowledge. More specifically, we deal with problems where the objectives of the multi-robot system can be transformed to the optimization of a specifically defined cost-function. Due to the unknown environment, unknown robots' dynamics, sensor nonlinearities, etc., the analytic form of the cost-function is not available a priori. Therefore, standard gradient descent-like algorithms are not applicable to these problems. In chapter 4, we first show that optimal one-step-ahead exploration schemes that are based on a transformed optimization criterion can lead to highly efficient solutions to the multi-robot exploration. As, however, optimal one-step-ahead solutions to the transformed optimization criterion cannot be practically obtained using conventional optimization schemes, the second step in our approach is to combine the use of the transformed optimization criterion with the Cognitive Adaptive Optimization (CAO): CAO is a practicably feasible computational methodology which adaptively provides an accurate approximation of the optimal one-step-ahead solutions. This combination results in a multi-robot exploration scheme which is both practically implementable

and provides with quite efficient solutions as it is shown both by theoretical analysis and, most importantly, by extensive simulation experiments and real-life underwater sea-floor mapping experiments in the Leixões port, Portugal. Finally, in chapter 5, we propose a distributed algorithm applicable to a quite large class of practical multi-robot applications. In such multi-robot applications, the user-defined objectives of the mission can be casted as a general optimization problem, without explicit guidelines of the sub-tasks per different robot. A novel distributed methodology is proposed, based on the CAO algorithm (as proposed on the previous chapter) that carefully designs a cost-function for each operational robot, where the joined optimization of which can accomplish the overall team objectives. The latter can be achieved by online learning (on each robot), only the problem-specific characteristics that affect the accomplishment of the overall mission objectives. The overall, low-complexity algorithm, can straightforwardly incorporate any kind of operational constraint, is fault tolerant and can appropriately tackle time-varying cost-functions. A cornerstone of this approach is that it shares the same convergence characteristics as those of block coordinate descent algorithms. The proposed algorithm is evaluated in four heterogeneous simulation set-ups under multiple scenarios, against both general purpose (centralized) and specifically-tailored to the problem in hand, algorithms.

Περίληψη

Η παρούσα διατριβή ασχολείται με το πρόβλημα της πλοήγησης ομάδων ρομπότ σε άγνωστα ή μερικώς γνωστά περιβάλλοντα, έτσι ώστε να καλυφθούν οι στόχοι της εκάστοτε αποστολής. Η δομή της διατριβής χωρίζεται σε δυο κύριους πυλώνες.

Ο πρώτος πυλώνας αφορά τον σχεδιασμό τροχών (offline) για περιπτώσεις στις οποίες υπάρχει πληροφορία σχετικά με το περιβάλλον που χρειάζεται να καλύψει η ομάδα από ρομπότ. Για την περίπτωση του ενός ρομπότ, όπου το πρόβλημα είναι γνωστό και ως Σχεδιασμός Τροχιάς για Κάλυψη (Coverage Path Planning, CPP), μια βέλτιστη O(n) μεθοδολογία έχει προταθεί, όπου n είναι το μέγεθος του πλέγματος που πρέπει να καλυφθεί. Δυστυχώς, όταν εμπλέκονται παραπάνω από ένα ρομπότ το πρόβλημα γίνεται NP-hard και μόνο προσεγγιστικές μεθοδολογίες έχουν προταθεί. Στο 3ο κεφάλαιο της παρούσας διατριβής, προτείνουμε έναν αλγόριθμο που χωρίζει τη διαθέσιμη περιοχή σε χωρικά-συμπαγείς υποπεριοχές, μία για κάθε ρομπότ. Αξίζει να σημειωθεί ότι οι αρχικές θέσεις των ρομπότ είναι μέρος της εξίσωσης και άρα δεν απαιτείται ξεχωριστός χρόνος έτσι ώστε να μεταφερθεί το κάθε ρομπότ στη δικιά του υποπεριοχή. Μετά από τη χάραξη αυτών των υποπεριοχών, εφαρμόζουμε κατανεμημένα τον βέλτιστο αλγόριθμο (STC), που έχει προταθεί για την περίπτωση του ενός ρομπότ, σε κάθε μια από αυτές τις περιοχές. Συνολικά, η μεθοδολογία πλοήγησης πετυχαίνει:

- να διασχίσει όλη τη διαθέσιμη περιοχή (complete coverage),
- περνώντας μόνο μια φορά από κάθε σημείο της περιοχής (without backtracking),
- πραγματοποιώντας ελάχιστα-ίδια μονοπάτια για κάθε διαθέσιμο ρομπότ (minimum coverage path per robot),
- και τέλος τα ρομπότ μπορούν να ξεκινούν από τις αρχικές τους θέσεις (initial positions constraint).

Μελετώντας τη σχετική βιβλιογραφία (κεφάλαιο 2), προκύπτει ότι καμία άλλη μέθοδος δεν πετυχαίνει όλα τα προηγούμενα χαρακτηριστικά στην παραγόμενη λύση της.

Ο δεύτερος άξονας αφορά την ανάπτυξη μια ομάδας από ρομπότ σε ένα τελείως άγνωστο περιβάλλον, έτσι ώστε να επιτευχθούν οι στόχοι της αποστολής. Στο δεύτερο άξονα οι αποφάσεις για την πλοήγηση των αυτόνομων οχημάτων λαμβάνονται σε πραγματικό χρόνο αξιοποιώντας τη γνώση (από τις μετρήσεις) που έχουν λάβει μέχρι το εκάστοτε βήμα. Η πλειονότητα των συγκεκριμένων προβλημάτων έχει αποδειχθεί αρκετά δύσκολη να επιλυθεί αποδοτικά. Στη βιβλιογραφία το παραπάνω πρόβλημα έχει αντιμετωπιστεί με τις ακόλουθες κλάσεις προσεγγίσεων:

- *Βέλτιστος έλεγχος ή τεχνικές δυναμικού προγραμματισμού*
- *Άπληστοι αλγόριθμοι*
- *Εκμάθηση παραμέτρων ελέγχου μέσω εκτεταμένων προσομοιώσεων (simulation-based)*

Στο 2ο κεφάλαιο παρουσιάζουμε συνοπτικά τις βασικές αρχές που διέπουν τη λειτουργία τους αλλά και τα επιτεύγματα και τις αδυναμίες που παρουσιάζουν η κάθε μια από αυτές.

Στο 4ο κεφάλαιο προτείνουμε μια μεθοδολογία που είναι σε θέση να σχεδιάζει τις τροχιές των ρομπότ αυτόματα σε πραγματικό χρόνο, έτσι ώστε να κατασκευάζεται ο χάρτης της περιοχής στον μικρότερο δυνατό χρόνο. Το συγκεκριμένο πρόβλημα έχει αποδειχθεί ότι είναι NP-complete, έτσι δεν μπορεί να λυθεί με βέλτιστο τρόπο. Στο ίδιο κεφάλαιο δείχνουμε ότι το συνολικό πρόβλημα μπορεί να αντιμετωπιστεί επαρκώς, εάν σχεδιαστεί μια συνάρτηση κόστους (κριτήριο απόδοσης) που περιλαμβάνει όρους που αφορούν συγκεκριμένες παραμέτρους και μετρικές του προβλήματος της χαρτογράφησης. Παρόλα αυτά, οι άπληστες μεθοδολογίες δεν μπορούν να εφαρμοστούν στον πραγματικό κόσμο αφού θα απαιτούσαν από την ομάδα των ρομπότ να κάνει ένα (μεγάλο) σύνολο από κινήσεις και ύστερα να αποφασίσει ποια είναι η αποδοτικότερη για να ακολουθήσει. Για να αντιμετωπίσουμε το συγκεκριμένο πρόβλημα προτείνουμε μια μεθοδολογία πλοήγησης που θα μπορεί να υλοποιηθεί σε ρομποτικά αυτόνομα οχήματα πραγματικού κόσμου. Η μεθοδολογία αυτή βασίζεται στον Γνωσιακό Προσαρμοστικό αλγόριθμο Βελτιστοποίησης (Cognitive-based Adaptive Optimization, CAO) και είναι σε θέση να προσεγγίζει τις λύσεις από τους άπληστους αλγορίθμους μέσω ενός πρακτικά υλοποιήσιμου συστήματος αποφάσεων, αφαιρώντας τη μη ρεαλιστική απαίτηση για πραγματοποίηση ενός συνόλου από εντολές ελέγχου πριν τη λήψη

της απόφασης. Ο προτεινόμενος αλγόριθμος ξεπέρασε την υπάρχουσα στρατηγική χαρτογράφησης σε μια σειρά από εκτεταμένες προσομοιώσεις, αλλά και όταν εφαρμόστηκε σε πραγματικά μη επανδρωμένα υποβρύχια οχήματα που βρίσκονταν στο λιμάνι Leixões του Πόρτο.

Στο 5ο κεφάλαιο προτείνουμε έναν κατανεμημένο αλγόριθμο γενικού σκοπού, που είναι σε θέση να πλοηγεί ομάδες από ρομπότ με σκοπό την επίτευξη των αυθαίρετα ορισμένων στόχων της αποστολής. Η συγκεκριμένη μεθοδολογία επεκτείνει τον αλγόριθμο που προτάθηκε στο προηγούμενο κεφάλαιο, για αυτό το λόγο παρουσιάζουμε και μια λεπτομερή σύγκριση της απόδοσης των δυο αλγορίθμων. Το κύριο χαρακτηριστικό που διαφοροποιεί τον παρόντα αλγόριθμο - εκτός από την κατανεμημένη φύση του - σε σχέση με αυτόν που προτάθηκε στο 4ο κεφάλαιο, είναι η ικανότητά του να χρησιμοποιεί αποδοτικά πληροφορία από προηγμένες αποφάσεις, με σκοπό την προσέγγιση της παραγώγου της συνάρτησης κόστους που πρέπει να βελτιστοποιηθεί σε κάθε αποστολή. Συνολικά, η προτεινόμενη μεθοδολογία έχει τα ακόλουθα πλεονεκτήματα:

(α) δεν απαιτεί γνώση από τις δυναμικές του συστήματος που καλείται να βελτιστοποιήσει,

(β) μπορεί να ενσωματώσει οποιουδήποτε είδους λειτουργικούς ή φυσικούς περιορισμούς,

(γ) έχει τα ίδια χαρακτηριστικά σύγκλισης με την οικογένεια των block coordinate descent (BCD) αλγορίθμων,

(δ) είναι ανεκτική στον θόρυβο,

(ε) μπορεί να χειριστεί επαρκώς προβλήματα πλοήγησης πολλαπλών ρομπότ, όπου οι στόχοι αλλάζουν κατά τη διάρκεια της αποστολής, και

(στ) μπορεί να υλοποιηθεί σε ενσωματωμένα συστήματα με περιορισμένες ενεργειακές δυνατότητες.

Ο προτεινόμενος αλγόριθμος δοκιμάστηκε σε τέσσερα διαφορετικά προβλήματα που αφορούν την πλοήγηση ρομπότ, με αρκετά διαφορετικά σενάρια, συγκρινόμενος με γενικού σκοπού αλγορίθμους αλλά και μεθοδολογίες ειδικά κατασκευασμένες για το εκάστοτε πρόβλημα.

Contents

List of Figures

List of Tables

1 Introduction

Contents

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 [3]. In the foreseeable future, the usage of a single robot will become obsolete, as there is an ever-increasing interest of multi-robot systems. The causality of this trend is outlined in the following three points. First, the recent *advantages in hardware and communications* allow the cooperative deployment of many affordable robots. Second, the use of multiple robots introduces *redundancy* which can be translated to mission speed-up and/or fault-tolerant characteristics (e.g. in cases of one or more robots faces a malfunction). Third, the utilization of multi-robot teams may tackle *problems that cannot be solved with a single robot* (e.g. complete space coverage). Robotic missions in which the multi-robot configuration can be more appealing include: surveillance in hostile environments (e.g. areas contaminated with biological, chemical or even nuclear wastes), law enforcement missions (e.g. border patrol), agriculture activities (e.g. soil sampling), cleaning missions (e.g. cleaning up an oil spill), etc.

In all the aforementioned missions, there are several factors that affect the performance of the team of robots. These are related with the technological limitations of the hardware which is used and the methodologies that process and fuse data to obtain valid conclusions related with the

actual robot performance. A key element of success in almost every mission is the ability to exploit (known terrain) or produce maps (unknown environment) by utilizing all the available resources.

Generally speaking, a multi-robot system can be characterized as a set of robots operating in the same environment. However, robotic systems may range from simple sensors, acquiring and processing data [4], to complex machines with several degrees of freedom, able to interact with the environment in fairly complex ways [5–7]. In this Thesis, we primarily focus on mobile platforms, equipped with sophisticated sensors and actuators, able to execute complex tasks. More precisely, we study the problem of multi-robot exploration/coverage in both known and unknown environments, for online and offline approaches where the mission objectives may vary from coverage to surveillance and mapping. In the upcoming sections we introduce the reader to the details of the problems as studied in this Thesis as well as an overview of the proposed methodologies, respectively. A literature review on the scope of the main pillars of this Thesis is presented in chapter 2.

1.1 Offline Multi-Robot Coverage Path Planning

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.). For the single robot case, also known as single robot coverage path planning (CPP), an $\mathcal{O}(\mathfrak{n})$ optimal methodology has already been proposed and evaluated in the literature, where $\mathfrak{n}$ 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 chapter 3, 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).

1.2 Online Multi-Robot Trajectory Generation

In this field of study, the objective is to online calculate the robots' trajectories so as to achieve the mission objectives.

In chapter 4, we propose a methodology which deals with the problem of autonomously navigate the robots so as to construct an accurate map of the unknown area in minimum time. Such a problem can be transformed into a dynamic optimization problem which, however, is NP-complete and thus infeasible to be solved in optimal manner. A usual attempt is to relax this problem by employing greedy (optimal one-step-ahead) solutions which may end-up quite problematic. In this chapter, we first show that optimal one-step-ahead exploration schemes that are based on a *transformed optimization criterion* can lead to highly efficient solutions to the multi-robot exploration. Such a transformed optimization criterion is constructed using both theoretical analysis and experimental investigations and attempts to minimize the "disturbing" effect of deadlocks and nonlinearities to the overall exploration scheme. As, however, optimal one-step-ahead solutions to the transformed optimization criterion cannot be practically obtained using conventional optimization schemes, the second step in our approach is to combine the use of the transformed optimization criterion with the Cognitive Adaptive Optimization (CAO): CAO is a practicably feasible computational methodology which adaptively provides an accurate approximation of the optimal one-step-ahead solutions. The combination of the transformed optimization criterion with CAO results in a multi-robot exploration scheme which is both practically implementable and provides with quite efficient solutions as it is shown both by theoretical analysis and, most importantly, by extensive simulation experiments and real-life underwater sea-floor mapping experiments in the Leixões port, Portugal.

Finally, in chapter 5 we propose a distributed algorithm applicable to a quite large class of practical multi-robot applications. In such multi-robot applications, the user-defined objectives of the mission can be casted as a general optimization problem, without explicit guidelines of the subtasks per different robot. Due to the unknown environment, unknown robots' dynamics, sensor nonlinearities, etc., the analytic form of the

optimization cost function is not available a priori. Therefore, standard gradient descent-like algorithms are not applicable to these problems. To tackle this, we introduce a new algorithm that carefully designs each robot's sub-cost function, where the optimization of which can accomplish the overall team objective. Upon this transformation, we propose a distributed methodology based on the CAO algorithm (as proposed on the previous chapter), that is able to approximate the evolution of each robot's cost function and to adequately optimize its decision variables (robot actions). The latter can be achieved by on-line learning only the problem-specific characteristics that affect the accomplishment of mission objectives. The overall, low-complexity algorithm, can straightforwardly incorporate any kind of operational constraint, is fault tolerant and can appropriately tackle time-varying cost functions. A cornerstone of this approach is that it shares the same convergence characteristics as those of block coordinate descent algorithms. The proposed algorithm is evaluated in four heterogeneous simulation set-ups under multiple scenarios, against both general purpose (centralized) and specifically-tailored to the problem in hand, algorithms.

2 State of the Art

Contents

2.1 Offline Multi-Robot Coverage Path Planning

Despite the fact that mCPP(multi-robot Coverage Path Planning) 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 Thesis, thus, in order to construct a more appropriate and homogeneous pool of alternative works, only publications that are in line with our problem formulation (section 3.2) 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 [8].

The authors in [9] transformed, for the first time, the single robot Spanning Tree Coverage (STC) Algorithm [10] 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) [11], 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 [12]. 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 [13], 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 [10], 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 i) to minimize the maximum distance from their initial positions and ii) 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.

2.1.1 Area division, for multi-robot tasks

Considering that the proposed approach (chapter 3) for the mCPP problem utilizes a mechanism that breaks down the operation area into robot-exclusive zones, in this subsection we present the dominant area division techniques, in order to assist multi-robot tasks - not limited to area coverage problem.

An interesting method that falls in this class, has been presented in [14]. The operation area is divided using sweep-line approach and in the sequel, each sub-area 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 [15], 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 [16]. 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 [17], 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 one hand, there is no specific guarantee about the optimality of the area division, while on the other hand the initial robots' positions are not taking into account.

The algorithm described in [18], 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[1]. 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. [19–21]), have been relied either on the Lloyd's [22] algorithm, with the known convergence properties [23] or on the Voronoi partitioning [24]. 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 formulation 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 [25] or to optimally monitoring a dynamic event with heterogeneous sensory interest (e.g. oil spill) [1]. 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.

2.2 Online Multi-Robot Trajectory Generation

We continue the review of the state-of-the-art in the direction of of online path planning trajectory generation for multi-robot systems. Mathematically speaking, multi-robot exploration, i.e., the on-line generation of robot trajectories so as to maximize accuracy and efficiency of the mission is an NP-hard [26–28] or, equivalently, a non-convex dynamic optimization problem. As a result, any attempt to generate the globally optimal solution is not possible to end up with a computationally and practically feasible solution.

2.2.1 Optimal control or dynamic programming approaches

To alleviate the above problem, many multi-robot approaches attack, instead of the original problem, a simplified version of it. In such a way,

[1] Mixed integer linear programming (milp) solver http://lpsolve.sourceforge.net/

a computationally feasible solution, utilizing *optimal control or dynamic programming* techniques, is possible to be constructed at the expense, of course, of sacrificing global optimality. For instance, to render the decision-making scheme computationally feasible, many methodologies [29–31] assumed relaxed or linearized version of the multi-robot problem. A usual assumption is that the robots operate in a discrete space where their actions and measurements can also take values from a finite discrete set of values [32, 33]. The exploitation of the above assumption can lead to remarkable results in the context of multi-robot tasks, presenting many real-life applications, e.g. [34]. Unfortunately, these strategies cannot be fully informed by the (usual occurring) continuous field measurements, while they can be computationally intractable for large state systems, e.g. a mobile robot operating in the real world often has millions of possible states [35]. Other multi-robot approaches, that fall in this class, adapt the assumption of perfect or sufficient knowledge of the dynamics of the overall multi-robot system, i.e. the dynamics of each and every robot along with their interactions with the other robots and the external environment [36, 37]. In such cases, the multi-robot problem can be seen to be equivalent to a standard optimization problem, where the robots' decision values are generated according to e.g., a gradient-descent or gradient-descent-like algorithm [38]. However, the requirement for perfect or sufficient knowledge of the overall dynamics renders the overall control design practically infeasible in many multi-robots applications, as they typically involve a large number of controllable variables with highly complex and uncertain dynamics [39–41].

2.2.2 Optimal one-step-ahead/Greedy approaches

Another well-investigated class of multi-robot approaches, is the *optimal-one-step-ahead* methodologies. In this family of approaches, the next robots' decision variables are chosen greedily, so as to optimize an appropriately defined cost function that is related to the problem in hand. For instance, in the domain of multi-robot exploration, a common practice is to choose the next robots' positions that maximize the expected information gain [42–44] or minimize the trace of the EKF error covariance matrix [45, 46]. Another subset of approaches [47, 48] optimizes a *quality function* that encloses, in the best possible way, the goals of the trajectory generation problem without any prior knowledge about the terrain morphology or the underlying dynamics. The trajectories are chosen so as to optimize this quality function, using in most of the cases *model-free Gradient-decent*-like

approaches [38]. Although, many of these approaches have been successfully evaluated in real-life multi-robot platforms, the majority of them suffer from the following drawbacks. First and foremost, the non-linearities may give rise to undesirable divergence (such as in cases where the noise does not follow the Additive Gaussian White Noise model). For example, it is usually considered that a robot can accurately estimate the position of an object or a point in the environment (landmark/cell) as soon as it perceives it. In most of the existing *optimal-one-step-ahead* approaches, this assumption allows in each timestamp the a priori calculation of the cost function as well as the robots' decision variables that greedily optimize such a cost function. Moreover, such an assumption is crucial for overcoming deadlocks (local minima) which are frequently encountered when greedy approaches are employed [49,50]. However, in real-world set-ups the above-mentioned assumption renders the existing approaches not practicable as it is not possible anymore to calculate in real-time the cost function as well as the locations of the robots that greedily optimize it. As a consequence, the employed techniques for avoiding or escaping deadlocks cannot be realized in the real-life robots either. Finally, the selection of the adequate cost function that provides an efficient solution to the multi-robot problem is not always trivial.

2.2.3 Simulation-based methodologies

On the other side of the spectrum, are the *simulation-based* multi-robot methodologies [51–53]. The idea behind these approaches is the following: first a parametrized decision-making mechanism is devised for generating on-line the robot decisions, with different choices for its parameters, leading to different decision-making mechanisms. Then, realistic simulations or similar tools are used in order to optimize the parameters of the decision-making mechanism. Thus, conceptually, many of the optimization computations that otherwise would take place on-line are "moved" off-line. The drawbacks of such approaches are that, first, the simulations need to cover a wide range of different realistic scenarios (and thus they may become "expensive") and, second, since the dimensionality of the optimization problem is quite high, a large number of parameters is needed in order to come up with an efficient decision-making mechanism.

2.2.4 Centralized vs. distributed

We close this sub-section by mentioning that, for most of the centralized approaches, in all three classes, it is not clear how they can be extended

to have a distributed nature. Furthermore, the majority of the distributed multi-robot algorithms (e.g. [39, 49, 50]) exploit application-specific dynamics, therefore their solutions cannot be generalized to a broader context. In other words, if the problem objectives or the dynamics are changed, most of the existing approaches must be redesigned from scratch, to adequately tackle the alternated problem.

3 Offline Multi-Robot Coverage Path Planning

Contents

3.1 Introduction

This chapter deals with the path planning problem of a team of mobile robots, in order to cover an area of interest, with prior-defined obstacles. In the literature, this problem is often referred to as *coverage path planning*

(CPP) problem, 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] [54], autonomous underwater vehicles [55,56], unmanned aerial vehicles [57], demining robots [58], automated harvesters [59], planetary exploration [60], search and rescue operations [61].

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 3.2), 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 [62].

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. [63–65]). One of the dominant approaches is the spanning tree coverage (STC) algorithm [10], 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 "narrower" 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 3.3 of the Thesis.

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

[1] irobot website http://www.irobot.com/

time has been proven to be NP-hard [66]. 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 2.1 for more details). Besides the optimality features that characterize a solution that have been already defined from the single-CPP, here in mCPP problem, the challenge of designing 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 chapter 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 [67] (usually referred as *allocate-then-decompose* or *decompose-then-allocate*), where the building and allocation of the tasks are tackled in a separated fashion [18], 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 approximately polynomial with respect to the (grid size $\times$ number of 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 number of robots. 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 [68] with the known convergence properties.

- During the second phase, and in a completely distributed manner, the paths inside each robot's cluster are designed by utilizing the STC algorithm.

The outline of this chapter is as follows. The mCPP problem is transformed into an optimization problem in section 3.2, introducing all the essential notation. In section 3.3 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 3.4. On the same section, are formally described the essential conditions of the optimal solution. In section 3.5 is proposed the DARP algorithm, with a comprehensive discussion about its performance. The complete scheme for the mCPP problem is outlined in section 3.6. As proof of concept, in section 3.7 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 3.8.

3.2 Multi-Robot Coverage Path Planning Formulation

For ease of understanding, it is assumed that the terrain to be covered is constrained within a rectangle[2] 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.

$$\mathcal{U} = \{x, y : x \in [1, \mathtt{rows}], y \in [1, \mathtt{cols}]\} \tag{3.1}$$

where $\mathtt{rows}, \mathtt{cols}$ 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 $n = \mathtt{rows} \times \mathtt{cols}$.

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:

$$B = \{(x, y) \in \mathcal{U} : (x, y) \text{ is } \textit{occupied}\} \tag{3.2}$$

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

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

$$\mathcal{L} = \mathcal{U} \setminus B \tag{3.3}$$

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:*

$$\|x_i - x_j\| + \|y_i - y_j\| \leqslant 1 \tag{3.4}$$

(Henceforth, we use $\|\cdot\|$ to denote the Euclidean norm $\|\cdot\|_2$)

As typical in many CPP and mCPP approaches (e.g. [8,9,13,66,69,70]), 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:

$$\chi_i(t) = (x_i, y_i) \in \mathcal{L},\ \forall i \in \{1, \ldots, n_r\} \tag{3.5}$$

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 i-th robot inside $\mathcal{L}$ is represented as $\chi_i(t_0)$.

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

$$\begin{aligned} \underset{X}{\text{minimize}} \quad & \max_{i \in \{1,\ldots,n_r\}} |X_i| \\ \text{subject to} \quad & X_1 \cup X_2 \cup \cdots \cup X_{n_r} \supseteq \mathcal{L} \end{aligned} \tag{3.6}$$

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

3.3 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 (3.6), the aforementioned single robot CPP can be defined as:

$$\begin{aligned} \underset{X_1}{\text{minimize}} \quad & |X_1| \\ \text{subject to} \quad & X_1 \supseteq \mathcal{L} \end{aligned} \tag{3.7}$$

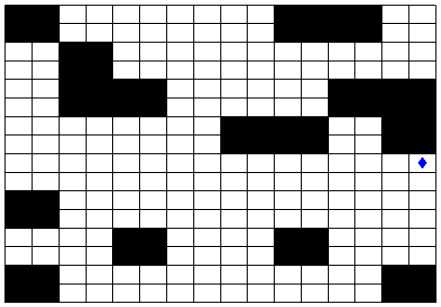
It has been proved in the literature that the CPP problem has an $\mathcal{O}(n)$ algorithm [10], where n is the size of grid, that is able to produce always the optimal solution. In other words, the Spanning Tree Coverage (STC) algorithm is able to construct the minimum path that coverages all the operation area $\mathcal{L}$, starting from any arbitrary unoccupied cell.

Figure 3.1 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 3.1(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 3.1(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 [71], as it is illustrated in Figure 3.1(c). The robot then *circumnavigates* the spanning-tree along a (counter) clockwise direction (Figure 3.1(d)). The circumnavigation of the spanning tree generates a simple *closed path* X_1^*, producing an optimal -in terms of coverage time- solution.

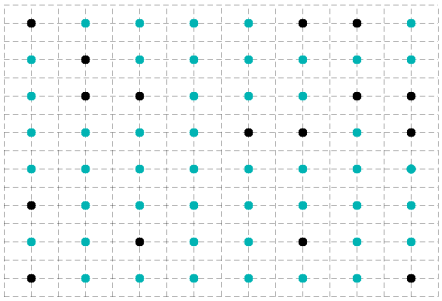
3.4 Reduce the original mCPP Problem

Utilizing the findings of STC algorithm for the case of one robot, the original mCPP problem, as defined in (3.6), can be reduced to

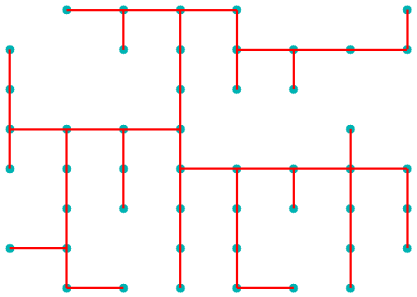
$$\begin{aligned} \underset{L}{\text{minimize}} \quad & \max_{i\in\{1,\dots,n_r\}} |L_i| \\ \text{subject to} \quad & L_1 \cup L_2 \cup \cdots \cup L_{n_r} \supseteq \mathcal{L} \end{aligned} \tag{3.8}$$



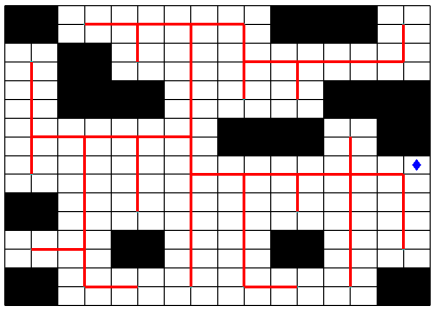
(a) Initial cells' discretization, robot's cell and obstacles



(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

Figure 3.1. *Spanning Tree Coverage Algorithm, sample execution*

where $L_1, L_2, \ldots, L_{n_r}$ 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 (3.7)). 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 regarding the L_i sets, so as the optimal solution to the overall mCPP (3.6) problem to be guaranteed.

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

1. $L_i \cap L_j = \emptyset, \forall i, j \in 1, \ldots, n_r \ , \ i \neq j$
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* $\forall i \in 1, \ldots, n_r$
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 (3.3) 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 seamless 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 (3.6).

Regarding to the existence of these solutions, it has been proved [72] 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

[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.

formulation of the problem along with proposed algorithm are referred to cases where at least, *one* optimal solution exists[4].

3.5 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 i-th 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 i-th robot's initial position $\chi_i(t_0)$. During each iteration, the assignment matrix A is constructed as follows:

$$A_{x,y} = \underset{i \in \{1,\dots,n_r\}}{\operatorname{argmin}} E_{i|x,y},\ \forall (x,y) \in \mathcal{L} \tag{3.9}$$

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

$$L_i = \{(x,y) \in \mathcal{L} : A(x,y) = i\},\ \forall i \in \{1,\dots,n_r\} \tag{3.10}$$

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

$$k_i = |L_i|,\ \forall i \in \{1,\dots,n_r\} \tag{3.11}$$

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.

[4]The interest reader is kindly referred to the appendix A for a proper classification of such cases where the optimality conditions cannot be met.

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.

3.5.1 Equally Divide the Space

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

$$E_{i|x,y} = d\left(\chi_i(t_0), [x,y]^\tau\right),\ \forall i \in \{1,\ldots,n_r\} \tag{3.12}$$

where $d(\cdot)$ denotes the chosen distance function (e.g. Euclidean). Thus, the initial assignment matrix A (3.9) 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:

$$E_i = m_i E_i \tag{3.13}$$

where m_i is a scalar correction factor for the i-th robot.

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

$$J = \frac{1}{2}\sum_{r=1}^{n_r} (k_i - f)^2 \tag{3.14}$$

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

A standard gradient descent method for updating m

$$m_i = m_i - \eta\frac{\partial J}{\partial m_i},\ \eta > 0,\ \forall i \in \{1,\ldots,n_r\} \tag{3.15}$$

can be employed, in an attempt to minimize the value of the cost function (3.14). When attempting to apply (3.15), 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 [68, Algorithm 1]. Coordinate descent algorithms solve optimization problems by successively performing approximate minimization along coordinate directions or coordinate hyperplanes. The global

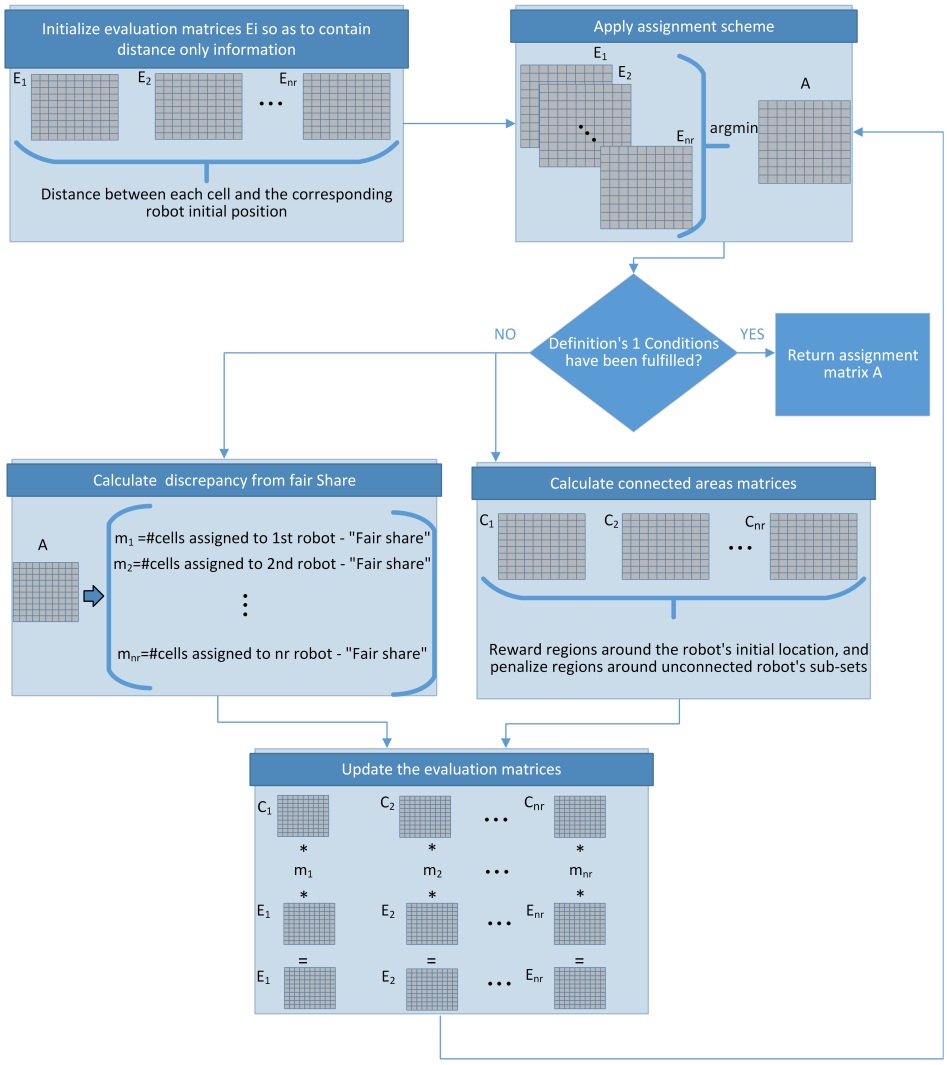


Figure 3.2. *DARP algorithm flowchart - Divide Areas based on Robots Initial Positions*

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 (3.14) can be obtained if we solve n_r single dimension optimization problems with the following objective function:

$$J_i = \frac{1}{2}(k_i - f)^2 \tag{3.16}$$

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 (3.16) are convex to the corresponding controllable parameter* m_i.

Proof. Let's assume that the i-th robot during the previous iteration, based on its evaluation matrix E_i, occupied less cells than the desirable threshold ($< f$). It is obvious from (3.13) and (3.9) 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 (3.16) will be decreased. Although, if we "over-decrease" the m_i factor "many" cells will be assigned to the i-th robot. Now, the J_i will start to rise again, as the k_i will be greater than the f. From this point and after, if we continue to decrease m_i, the i-th 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[5] ($l - n_r + 1$) have been assigned to the i-th 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 $J_i|_{k_i = l - n_r + 1} = \frac{1}{2}\left(\frac{(l-n_r)(n_r-1)}{n_r}\right)^2$. Therefore, the previously encountered minimum is the global one. The proof continues to hold if, instead, we had assumed that the i-th robot had been assigned to more cells than f. □

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

Additionally, the update rule of m_i can be straightforwardly calculated for each objective function (3.16) separately as:

$$\begin{aligned} m_i &= m_i - \eta \frac{\partial J_i}{\partial m_i} \\ &= m_i - \eta \left(k_i - f\right) \frac{\partial k_i}{\partial m_i} \end{aligned} \tag{3.17}$$

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 (3.16)). 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 (3.9). Therefore, the influence of $|\partial k_i / \partial m_i|$ can be securely omitted, and the final update policy can be approximated as follows:

$$m_i = m_i + c \left(k_i - f\right) \tag{3.18}$$

where c denotes a positive tunable parameter.

3.5.2 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 i-th robot that occupies more than one distinct regions the following matrix is introduced

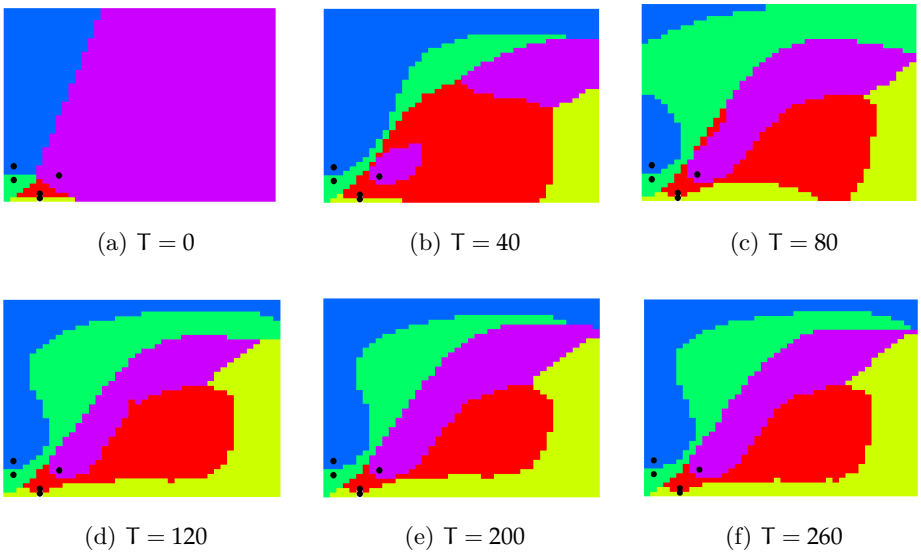
$$\begin{gathered} \mathcal{C}_{i|x,y} = \min\left(\|[x,y] - r\|\right) - \min\left(\|[x,y] - q\|\right), \\ \forall r \in \mathcal{R}_i,\ q \in \mathcal{Q}_i \end{gathered} \tag{3.19}$$

where $\mathcal{R}_i$ denotes the connected set of cells where the i-th 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 i-th 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 i-th 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 i-th robot belong to the same closed-shape region, the $\mathcal{C}_i$ is set to be the all-one-matrix.

The final update for the i-th evaluation matrix, is calculated as

$$E_i = C_i \odot (m_i E_i) \tag{3.20}$$

where $\odot$ denotes the element-wise multiplication. The findings of the previous subsections are illustrated in a flowchart format, in figure 3.2.



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

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

Figure 3.3. *Progression of the robots sub-regions over iterations*

3.5.3 Performance Discussion

Although simple in concept, the DARP algorithm aims to provide the optimal cells' assignment, in cases where at least one exists (according to Definition 3). A sample execution is illustrated in Figure 3.3, 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 (3.9) at the corresponding iteration. Apparently, the algorithm was terminated after 260 iterations, fulfilling all the conditions of Definition 3[6].

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

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

4.3(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* $\mathtt{i}$*-th robot assignment matrix*. Afterwards, the algorithm gradually eliminates the presence of unconnected areas, by reinforcing the robot's evaluation $\mathtt{E}_\mathtt{i}$ around the original (the one that the robot actually lies in) sub-area. By the time, the connectivity inside the exclusive robot sets $\mathtt{L}_\mathtt{i}$ is restored, the evaluation matrices $\mathtt{E}_\mathtt{i}$ will have completely changed their forms, and ideally towards to the optimal cells assignment.

Overall, 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.

3.5.4 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 $\mathcal{O}(\alpha \times n_r \times n)$[7] computational complexity. However, 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,

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

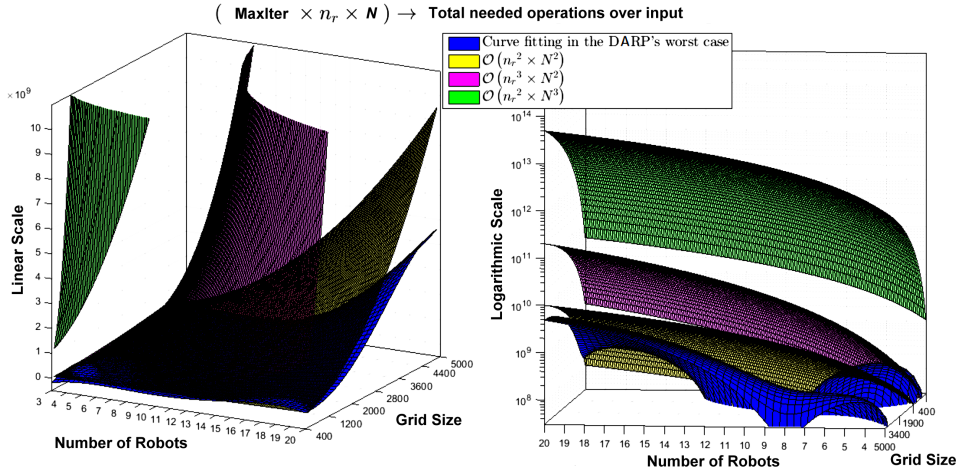


Figure 3.4. *Approximation on DARP's complexity, a comparison with known polynomial surfaces*

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 size of input parameters ($n_r \times n$). 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.

For each different $\{n_r, n\}$ scenario, the worst-case (maximum) of the needed iterations was extracted. In order to visualize the evolution of the worst-case computational needs (MaxIter) with respect to the input size, a polynomial least squares curve fitting technique is applied to these extracted values. The produced surface is illustrated with blue color in figure 3.4, 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 are 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 at most 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 of magnitude of the input, the algorithm may lose its polynomial behavior.

3.5.5 Beyond the classical mCPP

The proposed 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 3.2, 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 (3.16). More precisely, the objective function for the i-th robot is going to alternate as

$$J_i = \frac{1}{2}\,(k_i - p_i)^2 \tag{3.21}$$

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

However in order to be in-line with the ordinary mCPP formulation we limit our simulation evaluation (section 3.7) only to scenarios where an equal cells' division, between the robots, is considered desirable.

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

This section summarizes the complete algorithm for mCPP problem (3.6), 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 3.5. The outcome L_i of that process serves as the operational area for each robot separately (section 3.3).

After the applying of DARP algorithm and the corresponding production of L_i sets, the original multi robot optimization problem (3.6) is downgraded to n_r single robots CPP problems, alleviating its explosive combinatorial complexity [66]. Each one of these problems can be expressed as:

$$\begin{aligned} &\underset{X_i}{\text{minimize}} \quad |X_i| \\ &\text{subject to} \quad X_i \supseteq L_i \end{aligned} \tag{3.22}$$

where X_i denotes a robot path as defined in Definition 2. As shown in section 3.3 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. (3.6) problem, without any compromise on the quality or the generality of the solution. In essence, this can be attained by the original construction, during the first phase (section 3.5), 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.

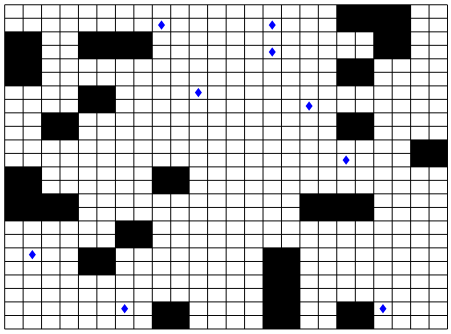
Figure 3.5 depicts an example execution of the proposed algorithm. Sub-figure 3.5(a) illustrates the initial robots' positions and the obstacles' placement. Sub-figure 3.5(b) visualizes each robot exclusive area, as calculated by the proposed DARP algorithm. Sub-figure 3.5(c) depicts the construction of the minimum spanning trees inside these divided areas. Finally, the proposed algorithm let the robots move along the path that circumnavigates the corresponding spanning tree, as is shown in sub-figure 3.5(d).

3.7 Simulation Results

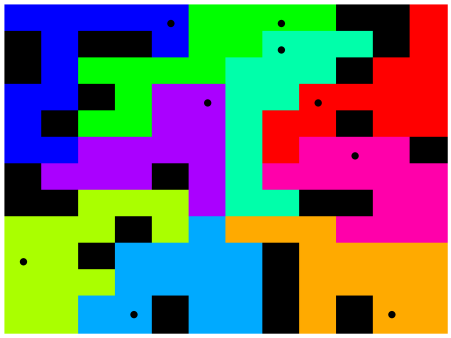
This section presents a comparison study between the proposed DARP+STC algorithm and two of the state-of-the-art methods ("MFS" and "Optimized MSTC" see related work). In order to produce comparable results, we adopt the same simulation set-up as in [12]. More precisely:

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

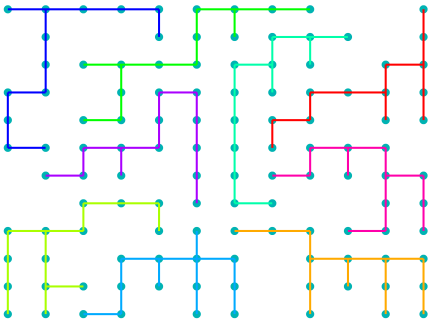
Table 3.1. *Cover time (in terms of path length) for DARP+STC, compared with MFC and Optimized MSTC*



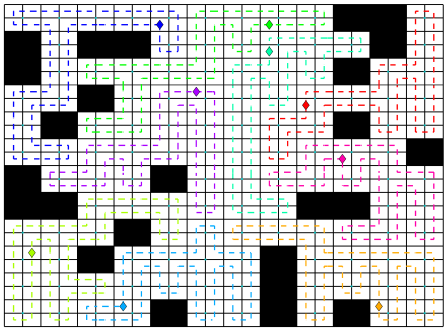
(a) Initial cells discretization, robots cell and obstacles



(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

Figure 3.5. *DARP+STC Proposed Approach, sample execution with 24x24 grid size, 9 robots and 100 obstacles*

- The size of the terrain was always $[\mathsf{rows}, \mathsf{cols}] = 98 \times 98$.
- We considered two kinds 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 followed 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 3.1, 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. 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.

A direct observation is that the performance of the proposed algorithm DARP+STC seems to be immune to the number of robots, obstacles and 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, i.e. $\big|\,|X_i| - |X_j|\,\big| \leqslant 4,\ \forall i,j \in 1,\ldots,n_r$. The above effectiveness bound is straightforwardly incoming from optimality guarantee of the proposed area division algorithm (DARP). 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 appendix A, where also preliminary solutions, in-line with the proposed approach, are also presented.

3.8 Conclusions

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 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.

4 Real-time Multi-Robot Exploration of Unknown Environments

Contents

4.1 Introduction

Recent technological advances have made the usage of teams of autonomous vehicles more appealing in a variety of missions [73], which may include harbor security [74–76], post-disaster infrastructure inspection [77, 78], underwater archaeology [79, 80], continuous infrastructure monitoring to prevent accidents [81], habitat mapping [82], spy missions, unmanned weapon and other military activities [83] as well as agricultural activities such as example paddy monitoring and spraying [84, 85]. In all the aforementioned missions there are several factors that affect the performance of the robot (in the case of a single vehicle) or of the overall team (in the case of a team of robots operating simultaneously). These are related with the technological limitations of the hardware which is used and the methodologies that process and fuse data to obtain valid conclusions related with the actual robot performance. A key element of success in almost every mission is the ability to produce valid maps by utilizing all the available resources.

There are, basically, two different problems that the team of robots faces when deployed in missions such as the ones mentioned above. The first of these problems has to do with the ability of the robots to process their sensor measurements so as they create accurate maps of the environment. As creation of accurate maps requires the robots to "know where they are", such a problem is also known as the Simultaneous Localization and Mapping (SLAM) problem, i.e., the problem of processing the robots' sensor measurements so as to simultaneously identify "where they are" and create the map of the external environment. The second of the problems has to do with "which trajectories the robots have to follow", i.e., the problem[1] of *trajectory generation* for the robots so as to maximize SLAM efficiency.

Most of the research work has concentrated on the problem of SLAM (in case of single-robots) or Cooperative SLAM (C-SLAM) in case where a team of robots is deployed. Very powerful SLAM and C-SLAM methodologies have been proposed recently and have successfully demonstrated in real-life situations [86–88]. Despite, however, these advances, the vast majority of missions rely on pre-specified robot trajectories. In other words, the trajectory the robot has to follow is designed off-line, before its actual deployment. As the robot is called to map a partially known or, in some

[1]The problem of multi-robot *trajectory generation for maximizing SLAM efficiency* is also referred in the literature as *exploration* or *optimal motion strategy*. In the rest of this chapter, these terms will be used interchangeably.

cases, a totally unknown area, off-line designing of the robot trajectories may become quite problematic: first of all, the off-line design is quite likely to "miss" areas of crucial information; moreover, it may lead the robot to "waste" time mapping areas of little information. For this reason, in practice, robot-based mapping is accomplished by employing a costly and tedious repetitive procedure: firstly, an original trajectory is designed off-line, the robot is then deployed and maps the area following to the off-line designed trajectory, then based on the created map a new trajectory is designed off-line again, the robot is deployed according to this new trajectory, and so on. Apart from the fact that such a procedure is costly and tedious, it renders prohibitive the deployment of robot in time-critical mapping missions or in cases where there are limited resources available, such as detection of sunken drums leaking chemicals or search-and-rescue missions. Most importantly, off-line generation of the robot trajectories cannot take advantage and exploit the cooperative capabilities in case a team of robots is employed. Typically, multi-robot deployment for mapping purposes employ again pre-specified trajectories with no or little interaction between the robots or, in the best case, the robots communicate with each other so as to improve their localization estimates and/or to make sure that they are moved in certain formation. However, full exploitation of the cooperative capabilities of a multi-robot system cannot be accomplished by having the robots moving along pre-specified trajectories or in formation: the cooperation between more than one robots can speed up considerably the overall mapping process, by having the robots coming closer in areas of high importance and by having the robots sharing sensor measurements and mapping information.

To overcome the shortcomings of off-line trajectory generation, many different approaches have been proposed which attempt to generate in real-time the robots' trajectories so as to maximize the overall C-SLAM efficiency, see chapter 2 for more details. There are, however, several theoretical and practical limitations that prevent these approaches from becoming a generic and practicable tool that will provide efficient trajectory generation: the fact that trajectory generation for maximizing SLAM efficiency is a difficult-to-be-solved optimization problem, the strong reliance of trajectory generation to the particular SLAM methodology employed, the highly non-linear nature of sensor noise and the limited communication capabilities of the robots are among the most important of such limitations. In this chapter, we propose and evaluate both using theoretical analysis and simulations as well as real-life experiments a new methodology that

attempts to overcome such limitations.

One of the most severe limitations of multi-robot trajectory generation for maximizing SLAM efficiency is the fact that such a problem is an NP-hard optimization problem. Most of the existing approaches employ one-step-ahead optimization or relaxed versions of the NP-hard trajectory generation optimization problem to overcome such a limitation. Such approaches, however, may end-up being quite problematic. First of all, the closed-form (i.e., analytical mathematical form) that relates the SLAM efficiency to the overall multi-robot team dynamics is not easy to be calculated. However, calculating of the analytical form of SLAM efficiency is the least of the problems encountered: the most severe problem is due to the fact that optimizing the SLAM efficiency may lead to severe *deadlocks* or, mathematically speaking, to getting stuck into local maxima. As a matter of fact, as we report in the next sections, one-step-ahead optimization of the SLAM efficiency can lead to situations where the robots get stuck to deadlocks even after they have accomplished only 10-20% of their mapping mission. A similar situation is also present if relaxations to the original NP-hard problem are employed. Moreover, any approach used for optimizing the SLAM efficiency has to deal with another problem: as, typically SLAM algorithms are based on linearized or approximate models for the sensor dynamics, optimizing the SLAM efficiency does not guarantee that poor performance or, even, divergence of the SLAM procedure is avoided.

To overcome all the above shortcomings and limitations, a new approach is employed and analyzed in this chapter. According to this new approach, the robots' trajectories are calculated so as to optimize a transformed version of SLAM efficiency: such a transformation guarantees that deadlocks are avoided and, moreover, that the robots move towards minimizing the effect of sensor non-linearities which, in turn, implies minimizing the problem of poor performance/divergence of the SLAM procedure. Theoretical analysis establishes these properties and, moreover, simulation experiments exhibit that the use of such a transformed version can improve significantly the performance of the exploration scheme.

As in the case of SLAM efficiency, it is difficult – if not possible – to obtain an analytical form for the transformed version of the SLAM efficiency. To overcome such a problem, we employ Cognitive Adaptive Optimization methodology (CAO), an adaptive optimization approach that does not require the availability of analytical forms of the function to be optimized [89–91]. It has to be emphasized that CAO – successfully im-

plemented to the problem of multi-robot optimal surveillance coverage [2] – is computationally simple and thus scalable. Both simulation experiments and theoretical analysis establish that the use of CAO, combined with the transformed version of the SLAM efficiency, guarantees efficient performance of the overall exploration.

Apart from the limitations of existing approaches that have to do with the complexity of the overall exploration problem, it has to be emphasized that most of these approaches suffer from other shortcomings that render their application in real-life quite difficult: for instance, some of these approaches impose certain requirements for the robot communication/sensing system and/or they strongly rely on a particular SLAM method. The development of the proposed approach is performed so as to avoid the above-mentioned shortcomings. As a matter of fact, rather than imposing requirements in the communication/sensing system and the SLAM methodology employed, the philosophy of the proposed approach is "to do the best it can" given the communication/sensing/ SLAM system, allowing even cases where the multi-robot team comprises robots with mutually different sensing capabilities or operating different SLAM algorithms. In such a way, the addition/removal of a robot (with probably different sensing capabilities than the existing ones or operating a different SLAM technique than the other robots) is performed "automatically".

A large worth-mentioned variation of this *optimal one-step-ahead*, class exploits the idea of the frontier cell navigation (e.g. [42, 50, 92–95]), firstly introduced by Yamauchi [96]. The majority of the one-step-ahead (greedy), path planning approaches utilize that fundamental idea and in the most cases they augment their solutions upon the frontier cell concept. Our approach is not an exception, as the omission of this concept would seriously compromise the overall successfulness of the mission. Over-and-above, the proposed approach introduces a supplementary improvement, regarding to frontier cell concept. In the proposed approach the next robots' positions are at most a predefined distance (which is directly related to the maximum possible speed of the robots). This constraint in robots' movement adds an extra layer of efficiency in the sense that it strictly bounds the waiting time for each robot, from the moment it reaches the desired waypoint until the new one. Additionally, and in correspondence to the majority of the members of this class, we further investigate the construction of the objective function (section 4.3), adding secondary terms in order to avoid undesirable deadlocked situations. But the development of another *optimal one-step-ahead* technique is not the main contribution of this chapter.

This chapter attempts to prove that even more complicated and unknown problems of the ones that belong to the target group of that class can be adequately addressed, problems where the objective/reward function is not a metric that can be computed a-priori. Overall, the theoretical contribution of this chapter should be classified as an *approximately optimal one-step-ahead approach* and in particular in the model-free class.

We close this section by mentioning that real-life underwater sea-floor mapping experiments were conducted using two AUV (Autonomous Underwater Vehicles) in the port of Porto, Portugal. The experiments exhibit the practicability and "ease-to-operate" of the proposed approach. Despite the fact that, the theoretical approach (section 4.2) along with the simulation results have been made under communication constraints, where at each time-step, every robot has to be at most a maximum distance from its closest robot [42], in real-life experiments, it is assumed that the robots' modems can transmit/receive from the whole operation area. This limitation is not unrealistic (at least until a specific order of magnitude in the square meters of the operation area), taking for granted that in the exhibited real-life experiments, the robots are able to transmit in each time-step their measurements back to central station. Additionally, the communication protocol that is used is well established in terms of the modern communication methods' [97–99] available bandwidth. The last remark regarding to the communications is that the theoretical approach along with the simulation results have been made independently on the place where the new waypoints calculation is taking place (either on a central command station or assuming a processor on one of the robots). This feature leverages the proposed approach with the ability of a broader appliance in robot's mapping missions.

The rest of the chapter is organized as follows. In section 4.2 we attempt to provide a description of the set-up of map construction using a team of robots. In section 4.3 we formulate the problem of autonomous navigation/exploration of a team of robots as an optimization problem, while in section 4.4 we present the proposed CAO based approach which allows the autonomous navigation/exploration of a team of robots. Finally details simulations that present the applicability of our approach are presented in section 4.5 while in section 4.6 we present how our approach was implemented as a plug-and-play tool for the navigation of a set of real autonomous underwater vehicles in the area of Oporto's Harbor in Portugal. Some concluding remarks and ideas for future research are discussed in section 4.7.

4.2 The Set-Up

In this section, we describe with the set-up assumed in this chapter, for map construction using a multi-robot team of N_R robots, where N_R denotes the number of robots.

4.2.1 Optimal Quantized Map

Without loss of generality, let us assume that the area to be mapped is constrained within a rectangle in the (x, y)-coordinates, i.e., the robots are called to map the area constrained in the (x, y)-coordinates as follows:

$$\mathcal{U} = \left\{ x, y : x \in [x_{min}, x_{max}], y \in [y_{min}, y_{max}] \right\}$$

where $x_{min}, x_{max}, y_{min}, y_{max}$ are real numbers that define the "borders" of the area to be mapped. Using the definition of $\mathcal{U}$, the area can be then defined as a function that corresponds each point $(x, y) \in \mathcal{U}$ to a point $z = z(x, y)$ [height of map at (x, y)]. Let us also consider a *fixed* set of N_L pairs $(x_\ell, y_\ell), \ell = 1, \ldots, N_L$ that are distributed in $\mathcal{U}$ [typically (x_i, y_i) are uniformly distributed in $\mathcal{U}$] and let $z(x_\ell, y_\ell)$ denote map's value at the point (x_ℓ, y_ℓ). Typically, the map construction problem can be transformed to the one of finding the "best grid" $(x_\ell, y_\ell, z_{L,\ell}), \ell = 1, \ldots, N_L$ such that the *quantized map*

$$z_q(x, y) = \sum_{\ell=1}^{N_L} z_{L,\ell} \phi_i(x, y, x_\ell, y_\ell)$$

approximates the actual map $z(x, y)$ as accurately as possible, i.e., the map construction problem is transformed into the problem of finding the parameters $z_{L,\ell}, \ell = 1, \ldots, N_L$ that minimize the following criterion:

$$\int_{(x,y)\in\mathcal{U}} \| z(x, y) - \sum_{\ell=1}^{N_L} z_{L,\ell} \phi_\ell(x, y, x_\ell, y_\ell) \|^2 \, dx dy \tag{4.1}$$

The functions $\phi_\ell(\cdot) \in [0, 1]$ in the above equation correspond to the so-called *basis functions*. Typical choices for $\phi_\ell(\cdot)$ is the piecewise constant function:

$$\phi_\ell(x, y, x_\ell, y_\ell) = \begin{cases} 1 & \text{if } \sqrt{(x - x_\ell)^2 + (y - y_\ell)^2} \\ & = \min_{j=1,\ldots,L} \sqrt{(x - x_j)^2 + (y - y_j)^2} \\ 0 & \text{otherwise} \end{cases} \tag{4.2}$$

or, the Gaussian function:

$$\phi_\ell(x, y, x_\ell, y_\ell) = \exp(-(x - x_\ell)^2 - (y - y_\ell)^2) \tag{4.3}$$

Let also

$$\mathbf{X}^L = \begin{bmatrix} x_1, y_1, z_{L,1}(x_1, y_1) \\ \vdots \\ x_{N_L}, y_{N_L}, z_{L,N_L}(x_{N_L}, y_{N_L}) \end{bmatrix}$$

Hereafter, we will define $\mathbf{X}^L$ to be the matrix of *map landmarks* associated with the map $z(x, y)$.

We close this section by mentioning that in the experiments described in this chapter, the basis functions were chosen to be the ones given in (4.3).

4.2.2 Robots Sensors

Before we continue on the map construction problem, let us first provide with some necessary preliminaries as far as the robot sensors are concerned. The robots are equipped with sensors that provide proprioceptive measurements (e.g., from GPS labels or inertial sensors) to propagate their state (position and orientation) estimates as well as exteroceptive measurements (e.g., cameras, sonars, bathymeters, etc) that enable them to measure their distance or bearing from points of interest. Let x_i^R denote the position (in a 3-D space) of the i-th robot [2] and

$$\mathbf{X}^R = \begin{bmatrix} x_1^{R^\tau} \\ \vdots \\ x_{N_R}^{R\ \tau} \end{bmatrix}$$

denote the matrix of all robots' positions (*team configuration*). Furthermore, let $\mathbf{Y}$ denote the vector of all robots' sensor measurements. In the most general case, the sensor measurements are related to the matrices $\mathbf{X}^L$ and $\mathbf{X}^R$ through a nonlinear function that admits the form

$$\mathbf{Y} = H(\mathbf{X}^L, \mathbf{X}^R, \Xi)$$

[2] For simplicity, we assume that the orientation of the robots is fixed and constant all the time. All the results of this chapter can be easily extended in the case where the orientation changes by the navigation algorithm.

where H is the nonlinear vector sensor function and Ξ is the sensor measurement noise vector.

The design for map construction using robots will have to take into account the – sometimes severe – limitations of the environment the robots operate on: nonlinear sensor noise characteristics, limited communication range, and limited visibility of the robots' sensors, are some of the limitations that render multi-robot map construction a very challenging task. Below, these limitations are described in more detail:

(NL-Noise) The typical assumption made in most robotic applications that the sensor noise is additive Gaussian noise is very restrictive and not realistic in many robot applications as the sensor noise affects the sensor measurements in a Non-linear fashion. For instance, in the case of sonar or vision sensors, the noise affecting such sensors is proportional to the sensor-to-sensing point distance, i.e., the larger is the robot-to-sensing point distance, the larger is the sensor noise. Similarly, in the case of localization-related sensors, the larger is the time the robot is operating without GPS recalibration, the larger is the localization noise. As a result, it is more realistic to assume a multiplicative sensor noise model that takes the form:

$$y = h(x^R, q) + h_\xi(x^R, q, t_{uw})\xi \tag{4.4}$$

where y is the sensor measurement, x^R, q are the positions of the robot and the sensing point, respectively, $h(x^R, q)$ is the sensor model in the noise-free case, $h_\xi(x^R, q, t_{uw})$ is the multiplicative sensor noise term, t_{uw} is the time the robot is without GPS connection and ξ is a standard Gaussian noise.

(LimComRange) In order to establish the required communication connection for the data transferring between the robots, an additional constraint is needed. For ease of comprehension, and without loss of generality, it can be considered that all robots have the same communication range, represented as com_R. As a result, a robot must make sure – at each time-instant – that there is at least one other robot that is within com_R distance from it.

(LimVis) In addition to the aforementioned limitations, the robot exteroceptive sensors are of limited visibility. As a result, additionally to the nonlinear sensor noise assumption (4.4), the sensor model for the exteroceptive sensors should be augmented to count for the limited visibility constraint. Moreover, the sensor model must be augmented to count for the case where there is no line-of-sight between the robot and the sensing point (e.g., there is an obstacle in between). As a result, the actual sensor

model becomes:

$$y_{x^R-q} = \begin{cases} \text{undefined} & \text{if } \|x^R - q\| \geqslant \text{thres} \\ \text{undefined} & \text{if there is no line-of-} \\ & \text{sight between } x^R \text{ and } q \\ h(x^R, q) + & \\ h_\xi(x^R, q, t_{uw})\xi & \text{otherwise} \end{cases} \tag{4.5}$$

where y_{x^R-q} denotes the sensor measurement from an robot at position x^R to a sensing point at position q, $\mathtt{thres}$ denotes the visibility threshold beyond which the sensor does not "see" and $\|\cdot\|$ denotes the Euclidean norm.

4.2.3 Aperiodic Robot Navigation/Communication under Communication Constraints

Having defined the limitations the robot sensors are facing, we now proceed by describing the way that the multi-robot team of robots is operating while performing the map construction task. More precisely, consider that the robots are initially deployed at the time instant t_0 with the robots initial positions being equal to $x_1^R(t_0), \ldots, x_{N_R}^R(t_0)$ [or, in a more compact notation, the overall multi-robot of robots initial position is $\mathbf{X}^R(t_0)$]. Then, by employing an autonomous navigation/exploration algorithm (to be described in next sections), the *next desired location* (waypoint) for each robot is calculated and each robot is navigated to this next desired location. Apparently, the time needed for each robot to reach its desired location is not the same for all robots. As a result, some of the robots will have to wait (after they have reached their desired location), so as for the rest of robots to reach their respective next desired location. Please note that while executing the task of navigating to the next desired location, the different types of robot sensors operate using different *activation frequencies*: for instance, the IMU sensors may be activated many times while the robots are accomplishing their navigation task, while the exteroceptive sensors, such as sonars, cameras and bathymeters are typically activated once and as soon as the robots reach and stabilize at their desired location. Let $\Delta t_{1,nav}$ denote the time required for *all* of the robots to reach their next desired location as well as to accomplish the activation of their exteroceptive sensors. Then, as soon as activation of all exteroceptive

sensors is accomplished, the robots communicate their sensor measurements – typically after some pre-processing – to a central station and receive back their next desired locations. Let $\Delta t_{1,com}$ denote the time required for all robots to communicate their sensor measurements and to receive back their next desired locations.

Using the above procedure we have that all of the robots have accomplished their navigation, sensing and communication tasks at the time-instant $t_1 = t_0 + \Delta t_{1,nav} + \Delta t_{1,com}$. At this time instant, the navigation/-exploration algorithm calculates the new desired locations for the robots and the above-described procedure is repeated again. In a nutshell, the robots receive at the time-instances $t_1, t_2, t_3, \ldots$ their new desired locations (waypoints) where $t_1 = t_0 + \Delta t_{1,nav} + \Delta t_{1,com}$, $t_2 = t_1 + \Delta t_{2,nav} + \Delta t_{2,com}$, $t_3 = t_2 + \Delta t_{2,nav} + \Delta t_{2,com}, \ldots$. As explained above, the time-intervals $[t_0, t_1)$, $[t_1, t_2)$, $[t_2, t_3), \ldots$ are not of the same length.

Please also note, that the overall exploration/ navigation process must be accomplished under *severe communication constraints*: due to these constraints, not all of the robot sensor measurements can be communicated to the rest of the team. Moreover, due to these constraints, one or more robots must not be able to communicate with the others (and, with the central station) at certain time-instances, which renders the overall exploration/navigation problem more challenging. For these reasons, the information to be communicated by each robot must be such that:

(C1) *It exploits to the maximum the cooperation capabilities of the system* by making sure that mapping information received by one robot that is useful to the other robots must be communicated.

(C2) Moreover, it must be able to efficiently operate when communication of one or more robots is lost at some time-instances or when the addition of a new member of the team is required "on-the-fly", i.e., when the team is in operation.

In the next sections, we will describe how we address these two issues that are related to the severe communication constraints of the operation environment.

4.2.4 Distributed Cooperative Estimation (SLAM) under communication limitations

The proprioceptive and exteroceptive sensor measurements are processed so as to perform both the map construction as well as the localization

of the robots. Accurate localization is a prerequisite for accurate map construction and, as a result, the overall estimation procedure requires the efficient simultaneous localization and mapping (SLAM). As already mentioned in the Introduction, there exists quite powerful methodologies for *single-robot SLAM*. Our approach is to develop a system that can take *any of these single-robot* approaches and appropriately enhance them so as to develop a *cooperative-robot SLAM* which will substantially increase the efficiency of single-robot SLAM by embedding within it with cooperative information while, of course, making sure that the two objectives (C1), (C2) related to the communication constraints are satisfied. More precisely, our approach comprises enhancing single-robot SLAM approaches by providing them with the "optimal" possible information received by the other robots. Below, we describe how such an approach is embedded within the proposed system.

Let us fix the number N_L of landmarks and let $\hat{\mathbf{X}}_L(0)$ denote an initial estimation of the map landmarks. Let also $\hat{\mathbf{X}}_R(0)$ denote the estimate of the initial robot positions. Then, in general, a single-robot SLAM system can be mathematically represented as follows:

$$\begin{aligned}&\left[\widehat{\mathbf{X}}_L^{(j)}(t_{i+1}), c_L^{(j)}(t_{i+1}), \hat{x}_R^{(j)}(t_{i+1}), c_R^{(j)}(t_{i+1})\right] = \\ &= \mathrm{EST}_j\left(\widehat{\mathbf{X}}_L^{(j)}(t_i), c_L^{(j)}(t_i), \hat{x}_R^{(j)}(t_i), c_R^{(j)}(t_i), y^{(j)}(t_{i+1})\right)\end{aligned} \tag{4.6}$$

where

- $\mathrm{EST}_j(\cdot)$ denotes the overall dynamics of the single-robot SLAM system for the j-th robot.
- $\hat{\mathbf{X}}_L^{(j)}(t_{i+1})$ denotes the estimated landmarks as generated by the single-robot SLAM system for the j-th robot.
- $c_L^{(j)}(t_{i+1})$ is an N_L-dimensional vector that corresponds to the *confidence level* of estimation of landmarks, i.e., the ℓ-th entry of the vector $c_L^{(j)}(t_{i+1})$ indicates the degree to how accurately the ℓ-th landmark has been estimated. For instance, in the case an Extended Kalman filter (EKF) is employed for performing the landmark estimation, the confidence level vector typically corresponds to the diagonal elements of the EKF error covariance matrix.
- $\hat{x}_R^{(j)}(t_{i+1})$ denotes the position estimate of the j-th robot.

- $c_R^{(j)}(t_{i+1})$ is a positive number indicating the confidence level of estimation of the robot position.

- $y^{(j)}(t_{i+1})$ is a vector comprising all sensor measurements that are available to the j-th robot.

Note that in case of single-robot missions, the sensor measurement vector $y^{(j)}(t_{i+1})$ comprises of the proprioceptive and exteroceptive sensor measurements of the single-robot. In the case, however, of cooperative robot missions, this vector can be augmented by using information that is communicated by the other robots. In other words, in the case of cooperative robot missions the sensor measurement vector $y^{(j)}(t_{i+1})$ can be augmented as follows:

$y^{(j)}(t_{i+1})$ = [*local sensor info of the* j*-th robot*, *sensor info from the other robots communicated to the* j*-th robot*]

The problem at hand becomes, then, to design the signals contained in *sensor info from the other robots communicated to the* j*-th robot* so as to substantially improve the estimator's (4.6) efficiency by fully exploiting the cooperation capabilities [objective (C1)] by taking into account the restrictions and possible malfunctions [objective (C2)] of the communication system. Before, we proceed on how to design the signals contained in *sensor info from the other robots communicated to the* j*-th robot*, we need some definitions, provided next.

Definition 4 We say that the ℓ-th landmark $X_\ell^L = (x_\ell, y_\ell, z_\ell)$ is visible if there exists at least one robot so that

- the robot and X_ℓ^L are connected by a line-of-sight;

- the robot and the point X_ℓ^L are at a distance smaller than the threshold value $\mathtt{thres}$ defined in equation (4.5) [which corresponds to the maximum distance the robots' exteroceptive sensors can "see"].

Given a particular team configuration $X^R(t_i)$, we let $\mathcal{V}(X^R(t_i))$ denote the set of all indices ℓ for which X_ℓ^L is visible at time-instant t_i. Similarly, we let $\mathcal{V}^{(j)}(X^R(t_i))$ to denote the set of all landmarks that are visible by the j-th robot.

In simple words, the subsets $\mathcal{V}^{(j)}(X^R(t_i))$ and $\mathcal{V}(X^R(t_i))$ provide us with information on the number of landmarks that are currently visible by the j-th robot and by the overall robot team, respectively.

Definition 5 We say that the ℓ-th landmark $\mathbf{X}_{\ell}^{L}(t_i)$ is accurately-estimated at the time-instant t_i, if the ℓ-th entry of the confidence level vector $\mathbf{c}_{L}^{(j)}(t_{i+1})$ of at least one robot is less than a given threshold[3]. Moreover, we define as $\mathcal{A}^{L}(t_i)$ the set of all indices ℓ for which $\mathbf{X}_{\ell}^{L}$ is accurately-estimated at time-instant t_i. Similarly, we define $\mathcal{A}^{(L,j)}(t_i)$ as the set of all indices ℓ for which $\mathbf{X}_{\ell}^{L}$ is accurately-estimated at time-instant t_i by the j-th robot, i.e., $\mathcal{A}^{(L,j)}(t_i)$ corresponds to the landmarks that have become accurately estimated due to the j-th robot.

Definition 6 We say that the ℓ-th landmark $\mathbf{X}_{\ell}^{L}(t_i)$ is inaccurately-estimated but visible at the time-instant t_i, if $\mathbf{X}_{\ell}^{L}(t_i) \in \mathcal{V}(\mathbf{X}^{R}(t_i))$ and $\mathbf{X}_{\ell}^{L}(t_i) \notin \mathcal{A}^{L}(t_i)$. Moreover, we define as $\mathcal{B}^{L}(t_i)$ the set of all indices ℓ for which $\mathbf{X}_{\ell}^{L}$ is inaccurately-estimated but visible at time-instant t_i. Similarly, we define $\mathcal{B}^{(L,j)}(t_i)$ as the set of all indices ℓ for which $\mathbf{X}_{\ell}^{L}$ is currently visible by the j-th robot but have not yet become accurately estimated.

Definition 7 We say that the robot $\mathbf{X}_{j}^{R}(t_i)$ is accurately-estimated at the time-instant t_i, if the confidence level $\mathbf{c}_{R}^{(j)}(t_{i+1})$ is less than a given threshold. Moreover, we define as $\mathcal{A}^{R}(t_i)$ the set of all indices j for which $\mathbf{X}_{j}^{R}$ is accurately-estimated at time-instant t_i.

In simple words, the subsets $\mathcal{A}^{L}(t_i), \mathcal{A}^{R}(t_i)$ provide us with information on the number of landmarks and robots, respectively, whose locations (positions) are currently accurately estimated, while the subset $\mathcal{B}^{L}(t_i)$ provides with information on the number of landmarks that are currently visible but have not yet been accurately estimated. Please note that, in the case of the subset $\mathcal{A}^{L}(t_i)$, if a particular index ℓ becomes a member of this set at some time, then it remains in this set for ever (once a landmark becomes accurately estimated, it remains accurately estimated forever). This is not true for the set $\mathcal{A}^{R}(t_i)$ as an robot whose position is currently accurately estimated may become inaccurately estimated later on. Moreover, please note that it suffices for at least one robot to estimate accurately a landmark. It is worth noticing that the aforementioned concepts are graphically illustrated in Figure 4.3.

We now return back to the problem of designing the signals contained in *sensor info from the other robots communicated to the* j*-th robot.* As

[3]Additionally, it might be useful to set an upper limit (big enough) in the times that a landmark can be estimated by at least one robot with any accuracy. This limit will serve as deadlock avoidance mechanism in cases of a landmark cannot be accurately estimated, due to the local morphology of the area to be mapped. We would like to thank one of the reviewers who pointed that out.

a first point, please notice that at the time-instant t_i, the j-th robot SLAM estimator may produce a number of new accurately estimated landmarks as well as a number of new visible but not accurately estimated landmarks. Mathematically speaking, during the aforementioned time-instant the j-th robot accurately estimates the landmarks that belong to the set $\mathcal{A}^{(L,j)}(t_i) \setminus \mathcal{A}^{(L,j)}(t_{i-1})$ while the landmarks that belong to the set $\mathcal{B}^{(L,j)}(t_i)$ are the ones that become visible but not accurately estimated by the j-th robot. In the ideal case of infinite communications resources, the following information should become available from the j-th robot to the rest of the team:

- The set of landmarks that have become accurately estimated at time-instant t_i by the j-th robot, i.e., the set $\mathcal{A}^{(L,j)}(t_i) \setminus \mathcal{A}^{(L,j)}(t_{i-1})$. Information of this set is needed by the other robots so as to know which landmarks have been accurately estimated by the j-th robots so as for them not to spend time estimating landmarks already accurately estimated. In other words, *this information is needed by the robots so as not to spend time exploring areas that have been already efficiently explored by some other robot.*

- The current position estimate $\hat{x}_R^{(j)}(t_i)$ of the j-th robot, the confidence level $c_R^{(j)}(t_i)$ as well as the sensor measurements that correspond to the $\mathcal{B}^{(L,j)}(t_i)$. Note that if this information were available to the other than the j-th robot, they could include in their SLAM scheme sensor measurements acquired by the j-th robot. In other words, if this information were available *each of the robots could use sensor information received by other robots as if it has been received by its own.* In such a way, the cooperative capabilities of the team could be exploited to the maximum extend.

As the team of robots has to operate under limited communication resources, exchanging the information described above is not possible. However, one can attempt to reduce the amount of the information described above by (i) tailoring this information so as it meets the communication requirements and (b) keep the information that can have the most significant impact to the SLAM estimators. This is done within the proposed system as follows:

- As a first step, for each robot, the members of the team that belong to the *reachable exploration area of the particular robot* are identified. The reachable exploration area for each robot is estimated as follows:

take any two robots, say the j-th and the k-th one. Let $\mathcal{P}$ denote the closest landmark [in the (x, y)-plane] to the k-th robot that is currently visible by the j-th robot. Then, the k-th robot belongs to the reachable exploration area of the j-th robot, if the distance in the (x, y)-plane between $\mathcal{P}$ and the position estimate of the k-th robot is less than a threshold which corresponds to the distance the k-th robot can travel in the time interval $[t_i, t_{i+1}]$. In other words, *if the k-th robot does not belong to the reachable exploration area of the j-th robot, then it is not possible for the k-th robot to "see" any of the landmarks currently "seen" by the j-th robot.*

- The landmarks that belong to the sets $\mathcal{A}^{(L,j)}(t_i) \setminus \mathcal{A}^{(L,j)}(t_{i-1})$ and $\mathcal{B}^{(L,j)}(t_i)$ are sorted in ascending order according to their distance to the robots that belong to the reachable area of the j-th robot. Then, the j-th robot communicates to the rest of the system its position estimate and the sorted landmark information up to the maximum communication capacity [in case where there are – estimations of – landmarks that are very close to each other, e.g., their distance is less than a pre-specified threshold, then only one of these landmarks is kept in the sorted list so as to avoid sending landmarks that contain similar information]. In return, the j-th robot receives its next way point, the position estimates of the robots that belong to its reachable exploration area as well as the sorted landmark information from the other robots up to the maximum communication capacity, where the sorting of the landmarks is done with respect to their (x, y)-distance from the j-th robot (and quantized by removing landmarks that carry similar information). By "landmark information" we mean (i) the index of the landmark in case this landmark belongs to the set $\mathcal{A}^L$ (i.e., in case it is an accurately estimated landmark), its estimated value $(\hat{z}_{L,\ell})$ and a binary signal indicating that it corresponds to an accurately estimated landmark (b) the index of the landmark and the respective sensor measurement in case this landmark is a currently visible but not accurately estimated landmark and a binary signal indicating that it corresponds to a visible but not accurately estimated landmark. Moreover, by "up to the maximum communication capacity" we mean that the packet to be sent or received is filled with as many landmarks as its maximum capacity allows.

By using the above logic for designing the signals to be communicated, useless sensor information is not communicated (i.e., information about

landmarks that it not possible for the robots to "see" in the next time-step as these landmarks are quite far from the current positions of the robots) while sensor information that can and should be used by the robots is quantized and prioritized by giving priority to landmarks that are more likely to be "seen" and, moreover, by removing information about landmarks that carry similar information.

We close this section by mentioning that in the simulation and experimental results detailed in later sections, the particular single-SLAM estimator used was a standard non-linear least-squares solver for estimating the landmarks and a cooperative EKF-based scheme for localization.

4.3 Autonomous Multi-Robot Robot Exploration as an Optimization Problem

The distributed SLAM procedure as described in the previous section can be successful only if the trajectories of the robots are designed in real-time, so as to optimize the estimators' performance. In other words, a real-time navigation procedure for the multi-robot system is needed which will optimize the performance of the overall estimation procedure. Apart from that, special emphasis must be given so as the navigation procedure is fault-tolerant, i.e., it is still working in case where one or more robot has lost connection to the rest of the team. In the next sections, we describe and analyze the navigation procedure developed within the proposed system.

4.3.1 Optimal Navigation/Exploration

By using all the preliminaries and definitions described previously, the optimal robot team navigation/exploration problem can be cast as a dynamic optimization problem as follows:

$$\max_{\mathbf{X}^R(t_1),\mathbf{X}^R(t_2),\ldots,\mathbf{X}^R(t_N),N} \sum_{i=1}^{N} J(t_i) \tag{4.7}$$

$$\text{s.t.} \quad C(\mathbf{X}^R(t_i)) \leqslant 0, i = 1,\ldots,N$$

$$J(t_i) = \frac{\left|\mathcal{A}^L(t_i)\right| - \left|\mathcal{A}^L(t_{i-1})\right|}{(t_i - t_{i-1})} \tag{4.8}$$

where N denotes the time-instant where all landmarks have been accurately estimated, $|\mathcal{A}|$ denotes the *cardinality* of the set $\mathcal{A}$ and $C(\cdot)$ is a non-linear function of the robot positions. The incorporation of the non-linear

function $C(\cdot)$ is used in order to constrain the robot waypoints $\mathbf{X}^R(t_i)$) to the rectangle $\mathcal{U}$ as well as to incorporate obstacle avoidance and maximum speed constraints. Standard algebraic manipulations can be employed to cast all these constraints in the form $C(\mathbf{X}^R(t_i)) \leqslant 0$. Note that the instantaneous cost $J(t_i)$ corresponds to the rate of number of landmarks that are accurately estimated per time unit. It is not difficult for someone to see that maximizing the criterion $\sum_{i=1}^{N} J(t_i)$ is equivalent of minimizing the time to accomplish the map construction procedure. It is worth noticing that the above formulation is not unique and other, different formulations can be applied as well. All the results of the proposed approach can be easily extended to the case where different formulations are used than the one above [by simply replacing the term $J(t_i)$ in equation (4.7) by the respective term of the different formulation]. Also, note that the accurate estimation of the robot positions (localization) is not directly used in the above formulation. However, accurate estimation of the robot positions is a prerequisite for accurate map construction and, as a result, the optimal solution of (4.7) requires that optimal waypoints $\mathbf{X}^R(t_1), \mathbf{X}^R(t_2), \ldots, \mathbf{X}^R(t_N)$ are chosen so that robots are optimally localized whenever it is necessary.

4.3.2 Optimal One-Step-Ahead Navigation/Exploration

There are many different reasons that render the solution of the dynamic optimization problem (4.7)-(4.8) practically infeasible. First of all, the complexity of the overall system dynamics renders practically very difficult to obtain closed-form (analytic) expressions for most of the terms in (4.7)-(4.8) which is a prerequisite, for applying most of the available tools from optimization theory. Most importantly and even if it were possible to obtain closed-form expressions, still the problem (4.7)-(4.8) would be impossible to be practically solved as it is a NP-complete problem. To overcome these two problems, we adopt a two step approach:

Step 1 In the first step, we assume that it is possible to obtain closed-form expressions for the terms in (4.7)-(4.8) and then attempt to modify the overall problem so as to come up with an approach that overcomes the NP-complete issue, on the one hand, but provides an efficient solution to the navigation/exploration problem, on the other. To do so, we adopt an approach where the problem is to construct – based on the dynamic optimization problem (4.7)-(4.8) – an appropriate objective function $\mathcal{J}(t_i)$ – different than $J(t_i)$ in equation (4.7) – so that *optimizing one-step-ahead* the function $\mathcal{J}(t_i)$ leads to an efficient solution to the navigation/exploration problem. In other

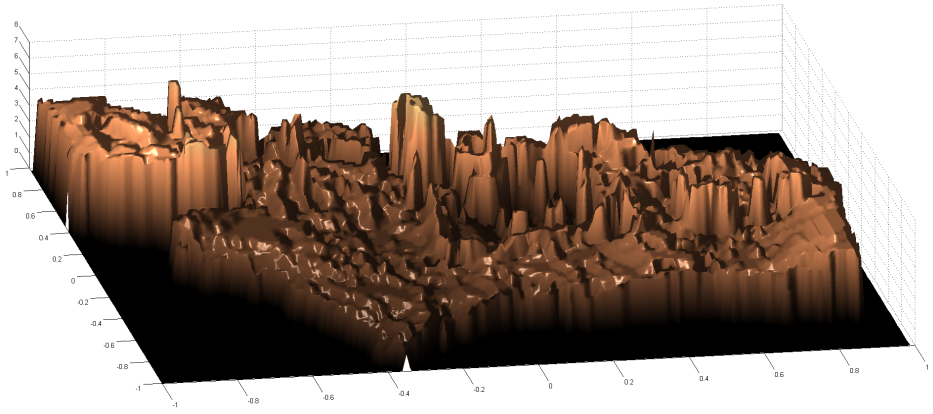
words, we attempt to construct a function $\mathcal{J}(\mathbf{t}_i)$ such that, choosing the waypoints $\mathbf{X}^R(\mathbf{t}_{i+1})$ that optimize $\mathcal{J}(\mathbf{t}_{i+1})$, provides an efficient solution to navigation/exploration problem. Both theoretical analysis and simulation experiments are used towards such a purpose. In the simulation experiments, to overcome the problem that closed-form expressions are not available, we employ the *Optimal-one-Step-Ahead Random Semi-Exhaustive Search – (OSARSES)*, an approach which approximates the optimal-one-step-ahead exhaustive search algorithm. Please note that OSARSES is a very "heavy" computational algorithm which cannot be implemented in real-life applications and is used only for analysis purposes.

Step 2 In order to overcome the problem that closed-form solutions of the terms involved in the optimization problem, we employ the Cognitive Adaptive Optimization (CAO) algorithm initially introduced in [89–91]. CAO has successfully been implemented to a similar – but significantly less complex – problem, this of navigating a team of autonomous robots when they are deployed to perform optimal surveillance coverage [2, 100, 101]. As in the case of navigation/exploration for map construction, the problem of optimal surveillance coverage involves optimization of terms for which closed-form expressions are not available. However, the problem of optimal surveillance coverage is a *static optimization* problem and, thus, less complex than the dynamic optimization problem of exploration treated here.

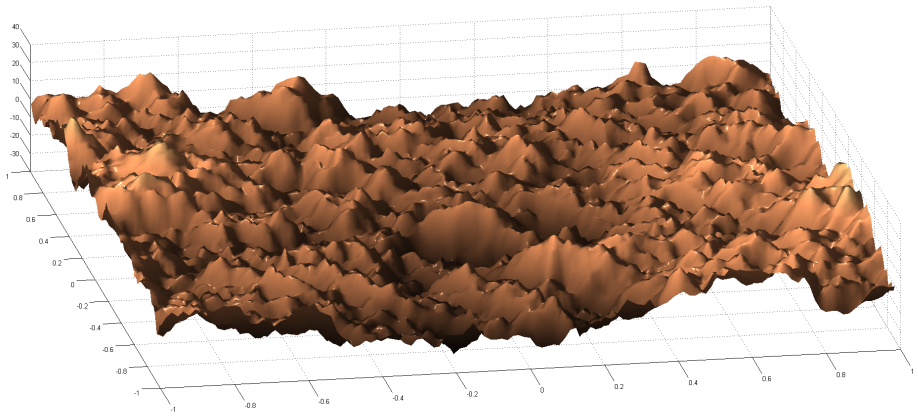
4.3.3 Transforming the Optimization Problem

An iterative approach was used in order to accomplish the first step described above:

Candidates for the objective function $\mathcal{J}$ were evaluated by employing multi-robot exploration using OSARSES for two different simulation environments. These two different simulation environments used two different quite complex maps (referred hereafter **Map #1** and **Map #2**). The first map (Map #1) depicts an area located in Zurich, Switzerland. This map was generated using a state-of-the-art visual-SLAM algorithm [100] which tracks the pose of the camera while, simultaneously and autonomously, building an incremental map of the surrounding environment. The second map (Map #2) is an artificially generated one, constructed using S-plines interpolation in such a way to represent sharp morphological variations in order to be used as worst-case mapping scenario. Both maps are illustrated in Figure 4.1



(a) Map #1



(b) Map #2

Figure 4.1. *The Simulation Environment*

For each of the candidate forms of $\mathcal{J}$ and for both of the simulation environments, the robots were navigated so as to optimize $\mathcal{J}$ by employing OSARSES: at each time-instant t_i, different sets of feasible candidate next waypoints $\mathbf{X}^{R,cand}(t_{i+1})$ for the robots were randomly generated and the overall system was simulated until the time-instant t_{i+1}; several such candidate sets were generated/simulated and the best [i.e., the one that provides the best $\mathcal{J}(t_{i+1})$] is chosen to be the next waypoints for the robots. The overall procedure of applying OSARSES is exhibited in Algorithm 1.

Using the above described procedure, an iterative design procedure was adopted for the design of $\mathcal{J}$: at each step of this iterative procedure,

Algorithm 1 One time-stamp of OSARSES Algorithm

Randomly Generate $\mathbf{X}^{R,cand}(t_{i+1})$
Exclude the ones that violates the environmental and communication constraints
$\mathbf{X}^{R,cand}_{valid}(t_{i+1}) \subseteq \mathbf{X}^{R,cand}(t_{i+1})$
$c \leftarrow 0$
while $c \leqslant |cand|$ **do**
  Deploy the action $\mathbf{X}^{R,cand(c)}_{valid}(t_{i+1})$
  $\mathcal{JJ}_c \leftarrow \mathcal{J}(t_{i+1})$ //Calculate the objective function value for this valid movement
  $c \leftarrow c+1$
end while
$imax \leftarrow \arg\max(\mathcal{JJ})$
$\mathbf{X}^{R,final}(t_{i+1}) \leftarrow \mathbf{X}^{R,cand(imax)}_{valid}(t_{i+1})$

the function $\mathcal{J}$ was modified, based on both observations on the system simulated performance and theoretical analysis.

In order to evaluate the performance under different choices for $\mathcal{J}$ we performed two different types of comparison. First, we compared the performance among these different choices as explained in the previous subsection. Secondly, we used an approximate solution to the dynamic optimization problem (4.7)-(4.8) and compared its performance with that of the proposed approach. More precisely, by employing the so-called Parametrized Cognitive Adaptive Optmization - (PCAO) - approach [102] was employed in order to approximately solve the dynamic optimization problem (4.7)-(4.8). By doing so, we concluded that the globally optimal – but practically not feasible – navigation/exploration algorithm for Map #1 can accomplish the overall map construction mission in less than 400 time-units, while the globally optimal navigation/exploration algorithm for Map #2 can accomplish the overall map construction mission in less than 450 time-units. It is worth noticing that the PCAO approach can provide a non-practically feasible solution as the number of computations required for implementing such a solution is huge. Moreover, as the exact solution to the dynamic optimization problem (4.7)-(4.8) is not possible to be obtained, these performance numbers correspond to upper bounds for the performance of the globally optimal solution.

However, as PCAO did not exhibit and any further significant improvement on these numbers as its approximation accuracy is increased, these upper bounds seem to be quite close to the performance bounds of the globally optimal performance. Having this in mind, all of the simulation experiments for both Map #1 and Map #2 were executed for 500 time-units and the performance index we adopted for evaluating the different choices

for $\mathcal{J}$ as well as the proposed CAO-based approach was the *map error at* $\mathtt{t} = 500$ (where by "map error" we define the percentage of landmarks that have not been accurately estimated). A solution that provides a small map error at $\mathtt{t} = 500$, although it is not the globally optimal, is very close to it.

Below, we describe the iterative procedure used for devising the objective function $\mathcal{J}$. Initially, the function $\mathcal{J}(\mathtt{t_i})$ was selected to be the instantaneous cost $\mathrm{J}(\mathtt{t_{i+1}})$ for the dynamic optimization problem (4.7)-(4.8). This is a typical choice in many different practical applications and it is the most straightforward choice for $\mathcal{J}$. However, the use of such a choice exhibits a very poor performance as shown in Table 4.1. More precisely and as shown in Table 4.1, this particular choice for $\mathcal{J}$ leads to a very poor performance (less than 15%[4] of the total number of landmarks are accurately estimated, for all different cases).

Landmarks	*1100*			*All*	
Candidates	**5**	**50**	**500**	**5**	**70**
Map #1	91.9%	88.9 %	85.3%	84.9%	85%
Map #2	87.5%	89.6 %	89.8%	87.8%	86.3%

Table 4.1. *Average percentage of Non-Accurately Estimated Landmarks at* $\mathtt{t} = 500$ *for* $\mathcal{J}(\mathtt{t_i}) = \mathrm{J}(\mathtt{t_{i+1}})$

The poor performance of the choice $\mathcal{J}(\mathtt{t_i}) = \mathrm{J}(\mathtt{t_{i+1}})$ is due to the fact that such a choice leads the overall algorithm to get stuck in *deadlocks* (or, mathematically speaking, the algorithm gets stuck in *local maxima*). To avoid such an unexpected situation, we modify the $\mathcal{J}(\mathtt{t_i}) = \mathrm{J}(\mathtt{t_{i+1}})$ by adding more terms, the maximization of which lead to a better performance. By employing the above-described iterative approach, the results of which are presented in the following subsections, the final form of the function $\mathcal{J}(\mathtt{t_i})$ is shown below:

$$\mathcal{J}(\mathtt{t_i}) = \mathrm{J}(\mathtt{t_{i+1}}) + \mathrm{J}_1(\mathtt{t_{i+1}}) + \mathrm{J}_2(\mathtt{t_{i+1}}) + \mathrm{J}_3(\mathtt{t_{i+1}}) \tag{4.9}$$

This enhanced version of the objective function retains only the terms that actually attribute to the general objective of the problem (see section 4.3.3 and figure 4.2). Below, we describe each of the terms in (4.9), as well as the reasoning behind choosing these terms.

[4]Table 4.1 presents the performance using the average percentage of the *Non-Accurately* estimated landmarks, so as to be in-line with the upcoming results.

Move towards the closest unexplored areas

The choice for the term J_1 is based on the observation that when there are no further landmarks that can be accurately estimated by some robots, the objective function can be augmented so as to "motivate" these robots to increase the number of *visible but non-accurately estimated* landmarks. The idea behind such an augmentation is simple: whenever some robots cannot estimate further landmarks, then they are "moved" to the closest unexplored areas. The particular form for J_1 that realizes such a reasoning is as follows:

$$J_1(t_i) = \sum_{j=1}^{N_R} s_{1,j}(t_i) \frac{\left|\mathcal{B}^{L,j}(t_i)\right|}{t_i - t_{i-1}} \tag{4.10}$$

where $\mathcal{B}^{L,j}(t_i)$ denotes the set of indices of visible, but non-accurately estimated landmarks by the j-th robot – see Definition 6 – and the term $s_{1,j}$ denotes a switching function that is zero when there are no deadlocks and becomes equal to a user-defined parameter K_1, otherwise:

$$s_{1,j}(t_i) = \begin{cases} 0 & \text{if } \left|\mathcal{A}^{L,j}(t_{i-1})\right| - \left|\mathcal{A}^{L,j}(t_{i-2})\right| \leqslant \epsilon_1 \\ K_1 & \text{otherwise} \end{cases} \tag{4.11}$$

where ϵ_1 is a small positive design constant. As it can be seen in Figure 4.2, the augmentation of the objective function with the term J_1 leads to dramatic improvements to the navigation/exploration task.

Avoid poor estimation

Additionally, to the augmentation using the term J_1, a further augmentation is used that takes into account the distances between every visible, non-accurately estimated landmark and its closest robot. More precisely, the objective function is augmented with the term J_2, which is used to force the robots to come closer to landmarks that are currently visible but not yet estimated, in order to complete their estimation process. Such a term takes into account the non-linear effect of the sensor noise [cf. equation (4.4)]: as the effect of sensor noise depends on the distance between the robot and the landmark, the estimation of the landmark becomes "better" when the robot moves closer to the landmark. Such an observation is realized by including the term J_2 defined as follows:

$$J_2(t_i) = -\sum_{j=1}^{N_R} s_{2,j}(t_i) \frac{\sum_{j \in B^L(t_i)} \min_{i=1,\ldots,N_R} y^2_{x_i^R - x_j^L}}{t_i - t_{i-1}} \tag{4.12}$$

where $s_{2,j}$ is a switching function defined similarly to $s_{1,j}$:

$$s_{2,j}(t_i) = \begin{cases} 0 & \text{if } \left|\mathcal{A}^{L,j}(t_{i-1})\right| - \left|\mathcal{A}^{L,j}(t_{i-2})\right| \leqslant \epsilon_1 \\ K_2 & \text{otherwise} \end{cases} \tag{4.13}$$

where K_2 is a positive design constant. In other words, the term J_1 is responsible for moving the robots closer to the landmarks that "have not seen before" (or "have been poorly seen"), while the term J_2 is responsible for moving the robots closer to landmarks so that they reduce the sensor noise effect and they can "see them better". Due to the tradeoff between the two terms, the constant K_1 and K_2 are used that serve as weights for giving less or more priority to one of the terms J_1, J_2.

The "curse" of full knowledge of a local region

After performing several experiments by using the cost criterion $\mathcal{J}(t_i) = J(t_{i+1}) +$
$J_1(t_{i+1}) + J_2(t_{i+1})$, it was observed that the use of such a criterion possessed the disadvantage that, in quite a few of instances, the robots get trapped in a sub-region that has been fully explored and there are no further non-accurately estimated landmarks that are or can become visible. As a result, any possible movement of the robots within the sub-region does not cause any change in the cost function. Consequently, the robots are "trapped" in a dead-lock. To avoid this undesirable situation, the term J_3 is introduced. Ideally, this term would attempt to minimize the distance between every robot and its closest landmark that is currently invisible and not accurately-estimated, so as to force the robots to move closer to landmarks that are non-visible and not-accurately estimated. As the calculation of this distance cannot be practically performed (as the exact position of this landmark is not known), firstly, the currently-invisible, non-accurately estimated landmarks *that have been visible at some time in the past* are examined. If no such a landmark exists (i.e., all the landmarks that were visible at some point, were accurately estimated), then the distances between the robots and the estimates of landmark positions is taking into account. Evidently, the term J_3 aims to move the robots in the "borders" between known and unknown areas and is defined as follows:

$$J_3(t_i) = -\sum_{j=1}^{N_R} s_{3,j}(t_i) \frac{\sum_{j \in \hat{\mathcal{B}}^L(t_i)} \min_{i=1,\ldots,N_R} y^2_{x_i^R - \mathbf{x}_j^L}}{t_i - t_{i-1}} \tag{4.14}$$

The set $\hat{\mathcal{B}}^{L}(t_i)$ denotes the set of indices of landmarks that were visible in the past but have not been accurately estimated and, if no such landmarks exist, the set $\hat{\mathcal{B}}^{L}(t_i)$ denotes the set of all landmark estimates that are or have not been visible and accurately estimated. The switching function $s_{3,j}$ becomes equal to 1 only when none of the terms J, J_1, J_2 can lead to a further improvement due to the j-th robot, i.e.,

$$s_3(t_i) = \begin{cases} 0 & \text{if(Condition <>)} \\ K_3 & \text{otherwise} \end{cases} \tag{4.15}$$

$$\text{Condition <>} = \left|A^{L,j}(t_{i-1})\right| - \left|A^{L,j}(t_{i-2})\right| \leqslant \epsilon_1$$

and

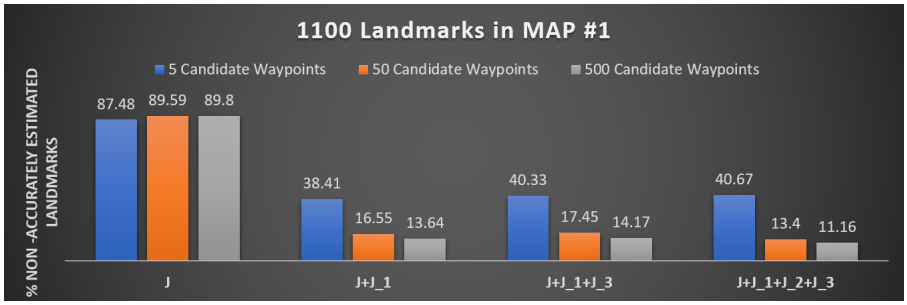
$$|J_{1,i}(t_{i-1}) - J_{1,j}(t_{i-2})| \leqslant \epsilon_2$$

and

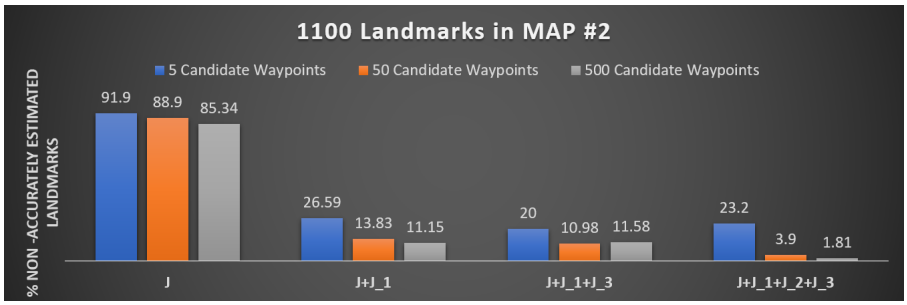
$$|J_{2,j}(t_{i-1}) - J_{2,j}(t_{i-2})| \leqslant \epsilon_3$$

Performance of the final form of the objective function

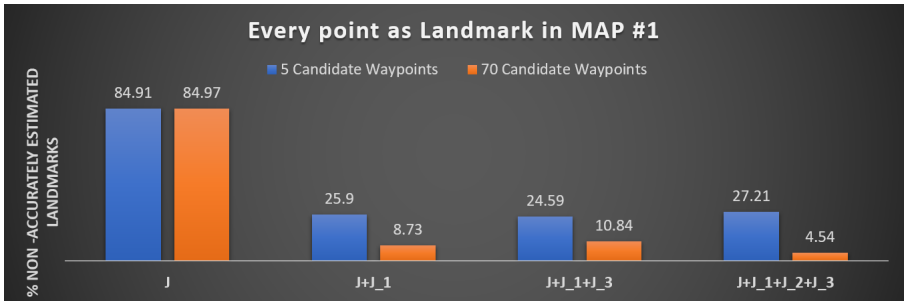
Figure 4.2 highlights the impact of each term of the objective function. It's worth noting that the simulation setup was kept the same for all the experiments, as explained in section 4.3.3. A conclusion that arises through the study of the results is the observation that the second term (see "Move towards the closest unexplored areas") strongly improves the overall performance. The rest of the terms are employed to guarantee the overall system's robustness, making it less vulnerable to the system's uncertain parameters such as the terrain's morphology and/or the initial arrangement of the landmark estimates. It is remarkable that in certain experiments, e.g. the one with 1100 landmarks on Map #1 with 500 candidate vectors for OSARSES, the exploration process managed to accurately estimate the 98.2% of the required landmarks, obtaining an improvement of 700% as compared to the choice $\mathcal{J} = J$. Summarizing, *the non-practically feasible method OSARSES under the choice (4.9) for the objective function can provide with quite efficient solutions under the condition that the number of candidate vectors for OSARSES is sufficiently large.*



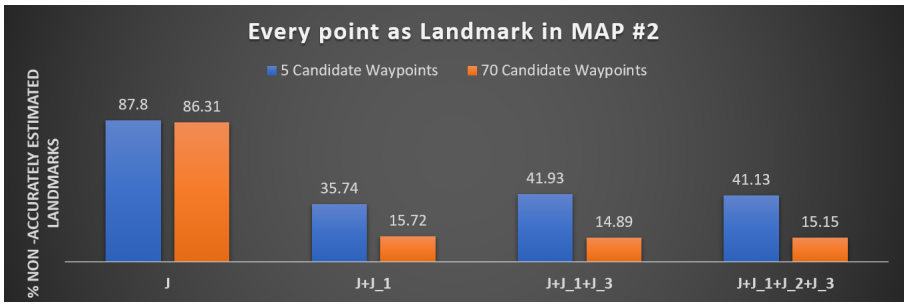
(a)



(b)



(c)



(d)

Figure 4.2. *Comparison of the average percentage of non-accurately estimated landmarks on 2 different maps and 2 different sets of landmarks*

Theoretical Analysis of $\mathcal{J}$

Apart from the numerical results that exhibit the efficient performance under the choice (4.9), it can be seen that such a choice leads to *deadlock-free* solutions and, moreover, the solutions obtained by one-step-ahead maximization this function, are efficient. Such a claim is formally stated in the next Theorem.

Theorem 1 *The objective function $\mathcal{J}$ as defined in (4.9) attains one and only maximum (i.e., it is free of local maxima). More precisely, its global maximum is attained when all the landmarks have been accurately estimated (i.e., when the overall map construction mission is successfully accomplished). Moreover, one-step-ahead maximization of this function guarantees efficient performance of the overall navigation/exploration mission. More precisely, let us consider the three different sub-teams of the overall robot team:*

- *the sub-team (A) which includes all robots of the team for which there are landmarks within their vicinity that can be accurately estimated;*
- *the sub-team (V) which includes all robots of the team for which there are landmarks within their vicinity that cannot become accurately estimated but they can become visible;*
- *the sub-team (I) which includes all robots of the team for which there are neither landmarks within their vicinity that can become accurately estimated nor landmarks that can become visible.*

Then, one-step-ahead maximization of $\mathcal{J}$ implies that (a) the sub-team (A) will be cooperatively navigated so as to maximize the number of landmarks to become estimated; (b) the sub-team (V) will be cooperatively navigated so as to maximize the estimation accuracy of the landmarks to become accurately estimated; (c) the sub-team (I) will be cooperatively navigated to the closest to each robot non-explored areas.

The proof of the above Theorem is straightforward as the functions $\mathrm{J}, \mathrm{J}_1, \mathrm{J}_2, \mathrm{J}_3$ were designed so as to satisfy the performance mentioned in the Theorem.

4.4 Cognitive Adaptive Optimization for Multi-Robot Exploration

Apart from the theoretical contribution about the construction of an objective function, the approximation of which will be sufficient enough

to navigate a team of robots in order to accurately map an unknown environment, our advantage against the majority of one-step-ahead optimal algorithms is that we utilize a more realistic approach where the actual evaluation of the objective function is not available at each timestep, only an approximation of it based on historical values. In essence, the actual objective/reward function at each timestep is highly depended on the morphology of the unknown (to be mapped) area and its evaluation without the actual movement should not considered trivial.

4.4.1 Preliminaries - Problem Conceptualization

Having defined the active exploration criterion, we will now proceed on presenting the proposed algorithm for autonomously navigating the robots towards maximizing such a criterion. The algorithm to be used is based on the so called Cognitive-based Adaptive Optimization (CAO) approach originated in the references [89–91]. CAO has been used in the past in a variety of robotics related applications, including implementations in aerial and ground robots, as described in detail in [2, 100, 103]. The version of the CAO algorithm used within the proposed approach, takes the same form and extends the one presented and formally analyzed in [2].

Below, we provide the main details of the CAO algorithm as employed in the framework of the active exploration problem. Please note that the only difference between the CAO approach used for the multi-robot optimal surveillance coverage problem [2, 100] and the one used here lies in the use of a different optimization criterion which, in turn, leads to different performance metrics (as detailed in Theorem 2 below).

We start by noticing that the active exploration criterion (equation 4.9) is a function of the robots' positions, i.e.,

$$\jmath_{t_i} = \mathcal{J}\left(\mathbf{X}^{R}_{t_i}\right) \tag{4.16}$$

where $t_1, t_2, t_3, \ldots$ denotes the time-index, $\jmath_{t_i}$ denotes the value of the active exploration criterion at the t_i-th time-step, $\mathbf{X}^{R}_{t_i}$ denote the position vectors of the robots (see 4.2.2), and $\mathcal{J}$ is a non-linear function which depends – apart from the robots positions – on the particular environment where the robots operate (e.g., position of landmarks).

Due to the dependence of the function $\mathcal{J}$ on the particular environment characteristics, the *explicit form of the function $\mathcal{J}$ is not known* in practical situations; as a result, standard optimization algorithms (e.g., steepest descent) are not applicable to the problem in hand. However, in most practical cases, like the one treated in this chapter, the current value of

the active exploration criterion can be estimated from the robots' sensor measurements. In other words, at each time-step t_i, an estimate of $\jmath_{t_i}$ is available through robots sensor measurements,

$$\jmath^{n}_{t_i} = \mathcal{J}\left(\mathbf{X}^{R}_{t_i}\right) + \xi_{t_i} \tag{4.17}$$

where $\jmath^{n}_{t_i}$ denotes the estimate of $\jmath_{t_i}$ and ξ_{t_i} denotes the noise introduced in the estimation of J_{t_i} due to the presence of noise in the robots' sensors. Please note that, although it is natural to assume that the noise sequence ξ_{t_i} is a stochastic *zero-mean* signal, it is not realistic to assume that it satisfies the typical Additive White Noise Gaussian (AWNG) property even if the robots sensor noise is AWNG: as $\mathcal{J}$ is a non-linear function of the robots positions (and thus of the robots sensor measurements), the AWNG property is typically lost.

Apart from the problem of dealing with a criterion for which an explicit form is not known but only its noisy measurements are available at each time, efficient robot navigation algorithms have additionally to deal with the problem of restricting the robots' positions so that obstacle avoidance and communication constraints are met. In other words, at each time-instant t_i, the vector $\mathbf{X}^{R}_{t_i}$ should satisfy a set of constraints which, in general, can be represented as follows:

$$\mathcal{C}\left(\mathbf{X}^{R}_{t_i}\right) \leqslant 0 \tag{4.18}$$

where $\mathcal{C}$ is a set of non-linear functions of the robots positions. As in the case of $\mathcal{J}$, the function $\mathcal{C}$ depends on the particular environment characteristics (e.g., location of obstacles, terrain morphology) and an explicit form of this function may be not known in many practical situations; however, it is natural to assume that the active exploration algorithm is provided with information whether a particular selection of robots' positions satisfies or violates the set of constraints (4.18).

Given the mathematical description presented above, the active exploration problem can be mathematically described as the problem of moving $\mathbf{X}^{R}_{t_i}$ to a set of positions that solves the following constrained optimization problem:

$$\begin{aligned} &\text{maximize} && \jmath_{t_i} \\ &\text{subject to} && \mathcal{C}\left(\mathbf{X}^{R}_{t_i}\right) \leqslant 0\,. \end{aligned} \tag{4.19}$$

As already noticed, the difficulty in solving, in real-time and in real-life situations, the constrained optimization problem (4.19) lies in the fact that

explicit forms for the functions $\mathcal{J}$ and $\mathcal{C}$ are not available. To circumvent this difficulty, the CAO approach of [90], appropriately modified the original CAO algorithm so as to be applicable to the problem in hand.

4.4.2 Main steps of CAO approach

Algorithm 2 One time-stamp of Cognitive Adaptive Optimization Algorithm

$\jmath^{n}_{t_{(i)}} - \mathbf{X}^{R}_{t_{(i)}}$ //update look-up/history tables with the previously evaluated pair
$\vartheta_{t_{i+1}} = \underset{\vartheta}{\operatorname{argmin}} \frac{1}{2} \sum_{\ell=\ell_{t_i}}^{t_i} \left(\jmath^{n}_{\ell} - \vartheta^{\tau} \phi\left(\mathbf{X}^{R}_{\ell}\right)\right)^2$ //Recalibrate the estimator's characteristics
Randomly Generate $\mathbf{X}^{R,cand}(t_{i+1})$
Exclude the ones that violates the environmental and communication constraints $\mathbf{X}^{R,cand}_{valid}(t_i) \subseteq \mathbf{X}^{R,cand}(t_{i+1})$
$c \leftarrow 0$
while $c \leqslant |cand|$ **do**
  $\hat{\jmath}_c \leftarrow \vartheta^{\tau}_{t_{i+1}} \phi\left(\mathbf{X}^{R,cand(c)}_{valid}(t_{i+1})\right)$ //Approximate the objective function value for every valid movement
  $c \leftarrow c + 1$
end while
$imax \leftarrow \arg\max(\hat{\jmath})$
$\mathbf{X}^{R,final}(t_{i+1}) \leftarrow \mathbf{X}^{R,cand(imax)}_{valid}(t_{i+1})$

As a first step, the CAO approach, which its pseudo-code is illustrated in Algorithm 2, makes use of function approximators for the estimation of the unknown objective function $\mathcal{J}$ at each time-instant k according to

$$\hat{\jmath}_{t_i}\left(\mathbf{X}^{R}_{t_i}\right) = \vartheta^{\tau}_{t_i} \phi\left(\mathbf{X}^{R}_{t_i}\right). \tag{4.20}$$

Here $\hat{\jmath}_{t_i}\left(\mathbf{X}^{R}_{t_i}\right)$ denotes the approximation/ estimation of $\mathcal{J}$ generated at the t_i-th time-step, ϕ denotes the non-linear vector of L *regressor terms*, ϑ_{t_i} denotes the vector of *parameter estimates* calculated at the t_i-th time-instant and L is a positive user-defined integer denoting the *size* of the function approximator (4.20). The vector ϕ of regressor terms must be chosen so that it is a universal approximator, such as polynomial approximators, radial basis functions, kernel-based approximators, etc.

The parameter estimation vector ϑ_{t_i} is calculated according to

$$\vartheta_{t_i} = \underset{\vartheta}{\operatorname{argmin}} \frac{1}{2} \sum_{\ell=\ell_{t_i}}^{t_{i-1}} \left(\jmath^{n}_{\ell} - \vartheta^{\tau} \phi\left(\mathbf{X}^{R}_{\ell}\right)\right)^2 \tag{4.21}$$

where $\ell_{t_i} = \max\{0, t_i - L - T_h\}$ with T_h being a user-defined nonnegative integer. Standard least-squares optimization algorithms can be used for the solution of (4.21).

As soon as the estimator $\jmath_\ell^n$ is constructed according to (4.20), (4.21), the set of new robots positions is selected as follows: firstly, a set of N candidate robots' positions is constructed according to[5]

$$x_{t_i}^{i,j} = x_{t_i}^{(i)} + \alpha_{t_i}\zeta_{t_i}^{i,j}, i \in \{1,\ldots,N_R\}, j \in \{1,\ldots,N\}, \tag{4.22}$$

where $x_{t_i}^{i,j}$ denotes the i-th element of $\mathbf{X}_{t_i}^R$, $\zeta_{t_i}^{i,j}$ is a zero-mean, unity-variance random vector with dimension equal to the dimension of $\mathbf{X}_{t_i}^R$ and α_{t_i} is a positive real sequence which satisfies the conditions:

$$\lim_{i\to\infty}\alpha_{t_i} = 0, \quad \sum_{i=1}^{\infty}\alpha_{t_i} = \infty, \quad \sum_{i=1}^{\infty}\alpha_{t_i}^2 < \infty\,. \tag{4.23}$$

Among all N candidate new positions $x_{t_i}^{1,j},\ldots,x_{t_i}^{N_R,j}$, the ones that correspond to non-feasible positions – i.e., the ones that violate the constraints (4.18) – *are neglected* and then the new robots positions are calculated as follows:

$$\left[\mathbf{X}_{t_i+1}^R\right] = \underset{\substack{j\in\{1,\ldots,N\}\\ \mathbf{X}_{t_i}^{R,j}\text{not neglected}}}{\operatorname{argmax}} \hat{\jmath}_{t_i}\left(\mathbf{X}_{t_i+1}^{R,j}\right)$$

The idea behind the above logic is simple: at each time-instant a set of many candidate new robots' positions is generated. The candidate, among all feasible ones, that provides the best estimated value $\hat{\jmath}_{t_i}$ of the objective function is selected as the new set of robots positions. The random choice for the candidates is essential and crucial for the efficiency of the algorithm, as such a choice guarantees that $\hat{\jmath}_{t_i}$ is a reliable and accurate estimate for the unknown function $\mathcal{J}$; see [90, 91] for more details. On the other hand, the choice of a slowly decaying sequence α_{t_i}, a typical choice of adaptive gains in stochastic optimization algorithms is essential for filtering out the effects of the noise term ξ_{t_i} [cf. (4.17)]. The next summarizes the properties of the CAO algorithm described above; the proof is among the same lines as this of Theorem 1 of [2].

[5] According to [90, 91] it suffices to choose N to be any positive integer larger or equal to $2\times$[the number of variables being optimized by CAO]. In our case the variables optimized are the robot positions $\mathbf{X}_{t_i}^R$ and thus it suffices for N to satisfy $N \geqslant 2N_R \times \dim\left(\mathbf{X}_{t_i}^R\right)$.

Theorem 2 *Let* $\mathbf{X}_{t_i+1}^{R,opt}$ *denote the "one-step-ahead-optimal" robot waypoints, i.e., the feasible waypoints that maximize* $\mathcal{J}(t_i)$*. Then, the above-described CAO algorithm satisfies:*

$$\mathbf{X}_{t_i+1}^{R} = \mathbf{X}_{t_i+1}^{R,opt} + \epsilon(t_i) + \nu$$

where $\epsilon(t_i)$ *vanishes to zero* exponentially fast *and the term* ν *is a constant term that depends on the approximator* ϕ *(and can become as small as desired at the expense of making the convergence of* $\epsilon(t_i)$ *slower).*

In simple words, the above Theorem states that the CAO-based approaches become (after some time due to learning) approximately equal to the optimal-step-ahead ones.

4.5 Simulation Results

In this section, we describe the simulation set-up used for the analysis presented in section 4.3.3 as well as the evaluation of the proposed CAO-based approach as compared to the practically infeasible OSARSES-based approach. The simulation environment for the experiments is described below:

- Simulations conducted using the two different maps described in detail in 4.2.4 and presented in 4.1. For simplification the operation area is restricted in the cube $[-1, 1]^3$, so any value, that is afterwards mentioned, including the maps, has been casted to this cube.
- The number of robots is equal to $N_R = 3$ for the first set of experiments (Figure 4.5) and $N_R = 10$ for the second (Figure 4.6).
- For the communication capacity, we assumed that each robot can send and receive up to 10 landmark measurements according to the procedure described in section 4.2.4.
- For the number of landmarks two different scenarios were evaluated. First, we assume that each map consists only of $N_L = 1100$ landmarks. In the second scenario, we assume that every point (pixel) of the height-map is a landmark. In this case, Map #1 includes $N_L = 7542$ landmarks while Map #2 includes $N_L = 10500$ landmarks.
- The main constraints imposed to the robots are that they remain within the terrain's limits, i.e., within $[X_{min}, X_{max}]$ and $[Y_{min}, Y_{max}]$

in the x- and y-axes, respectively. At the same time, robots remain within $[z + d, z_{Max}]$ along the z-axis, in order to avoid hitting the terrain. The value of d was equal to 0.05.

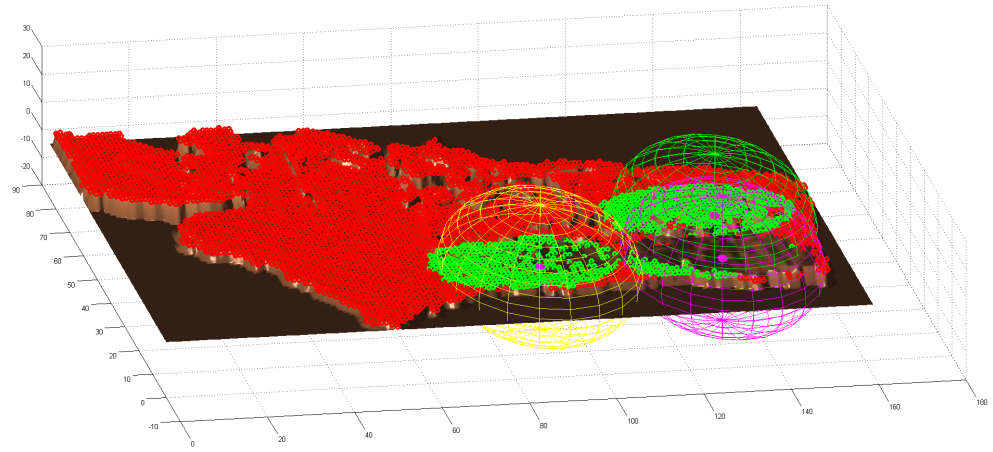
- The communication range is set to $com_{Range} = 0.3$.
- All robots were assumed to have range sensors, measuring the robot distance from the landmark, using the following equation:

$$y_{x^R - q} = \begin{cases} \text{undefined} & \text{if } \|x^R - q\| \geqslant \text{thres} \\ \text{undefined} & \text{if there is no line-of-sight between } x^R \text{ and } q \\ \|x^R - q\|\,(1 + \xi) & \text{otherwise} \end{cases} \tag{4.24}$$

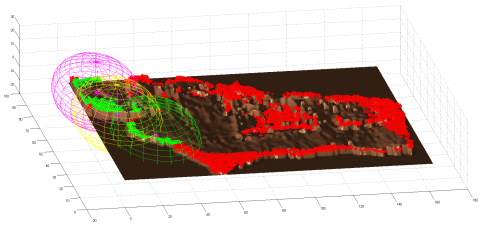
- Standard weighted least-squares [104] was employed for estimated the landmarks, while perfect localization accuracy was assumed.
- As an overall evaluation criterion for the exploration procedure (Figures 4.5(a), 4.5(c), 4.6(a) and 4.6(c)), the number of the *Remaining Landmarks*, ie the total number of landmarks that are not accurately-estimated after the completion of the experiment, will be used. Additionally to that term, and in order to obtain a better picture about the system's performance, we employed an extra evaluation criterion. The objective of this term is to distinguish between experiments where their final total number of the *Remaining Landmarks* was the same but their estimation progress was not. This criterion rewards performances that have greater landmarks' estimation ratio over the simulation time, ensuring to reward performances that achieved a satisfactory function from their early steps of the execution. This parameter (Figures 4.5(b), 4.5(d), 4.6(b) and 4.6(d)) corresponds to the summation of the error in the estimation of total landmarks from the first time-step to the last one, according to the equation 4.25

$$\text{Sum\_of\_Error} = \sum_{i=1}^{T_{max}} \|\hat{\mathbf{X}}^L(t_i) - \mathbf{X}^L(t_i)\| \tag{4.25}$$

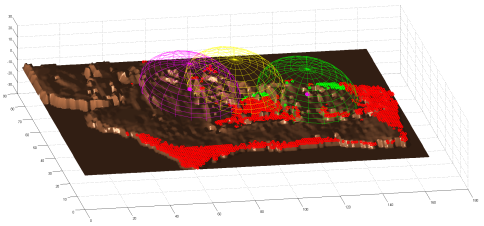
A set of experiments has been conducted in order to evaluate the performance of the proposed approach. One instance of the above experiments,



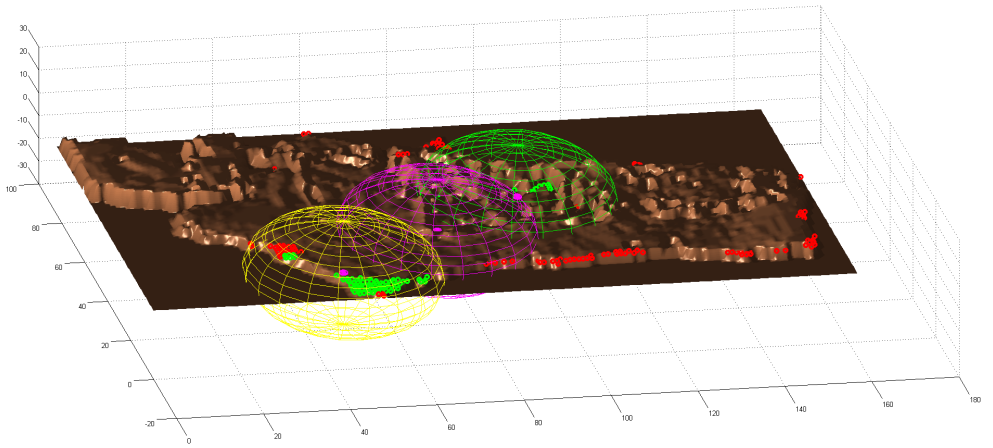
(a) T = 50



(b) T = 200

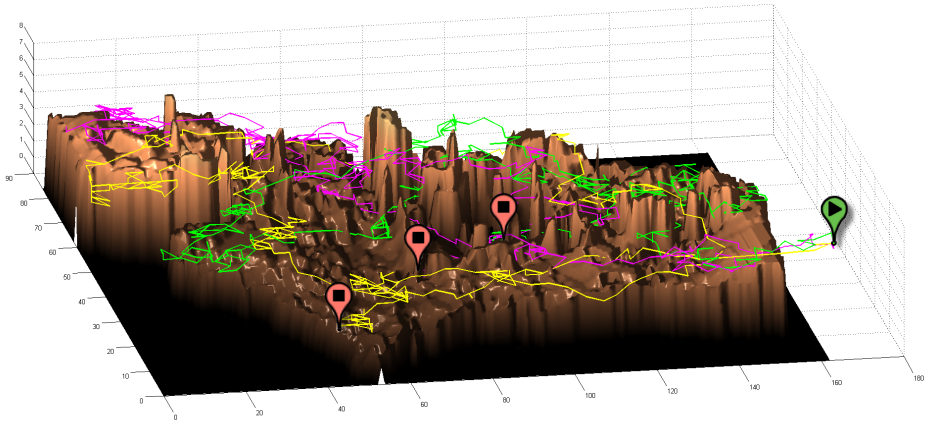


(c) T = 350

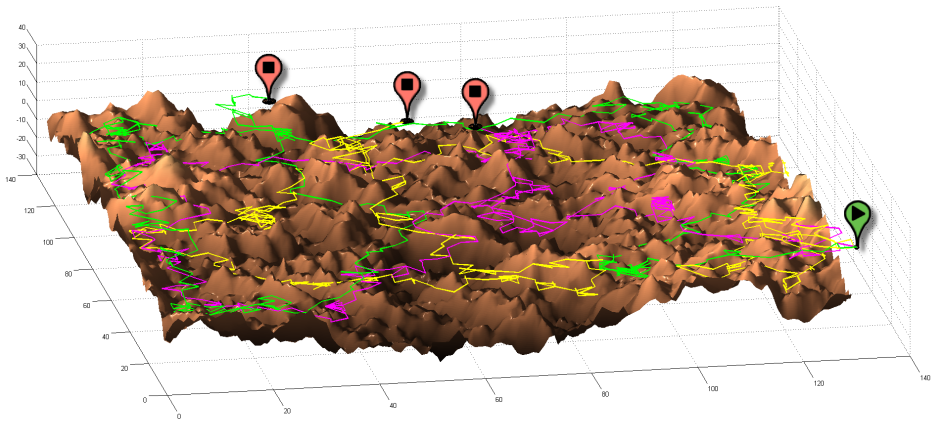


(d) T = 500

Figure 4.3. *Multi-robot Navigation/Exploration Process: the green area corresponds to currently visible landmarks, the brown area (with morphological characteristics) corresponds to landmarks that have been accurately estimated and the red area corresponds to landmarks that have never been seen before. The three big spheres indicate the communication range of each robot (located at the center of the spheres)*



(a) The Trajectories Recorded During the Navigation on Map #1



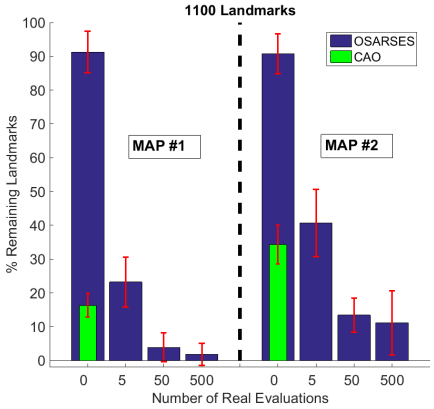
(b) The Trajectories Recorded During the Navigation on Map #2

Figure 4.4. *The Recorded Trajectories*

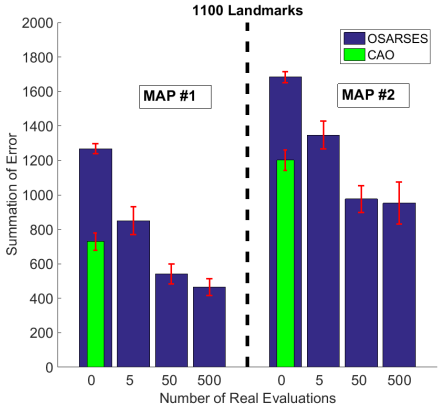
using 3 robots, has been illustrated in details, in Figure 4.3, where the green area corresponds to currently visible landmarks, the brown area (with morphological characteristics) corresponds to landmarks that have been accurately estimated and the red area corresponds to landmarks that have never been seen before. The three big spheres indicate the communication range of each robot (located at the center of the spheres). Figure 4.4 depicts the trajectories of the 3 robots in both maps, with their starting and ending positions, as they were calculated by the proposed CAO approach[6].

[6]A video footage of this experiment can be found on `https://tinyurl.com/`

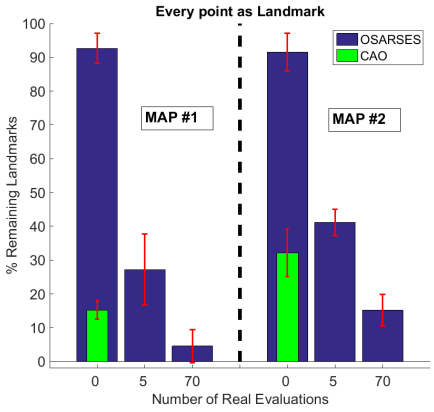
Overall the results are presented in Figure 4.5 for 3 robots and Figure 4.6 for 10 robots. The results from OSARSES are marked with the blue bars while the corresponding results of the CAO-based proposed approach with green ones. In each figure, the x-axis represents the number of real configurations that are evaluated at each timestep from the robots, before the final movement selection. It is worth highlighting that, the proposed approach is only located on the bar that corresponds to 0 real evaluations, as it does not need any actual movement in order to be able to make its decisions (see section 4.4).



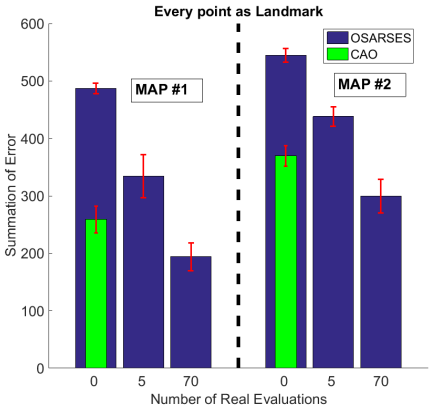
(a) %Remaining Landmarks



(b) Summation of Error



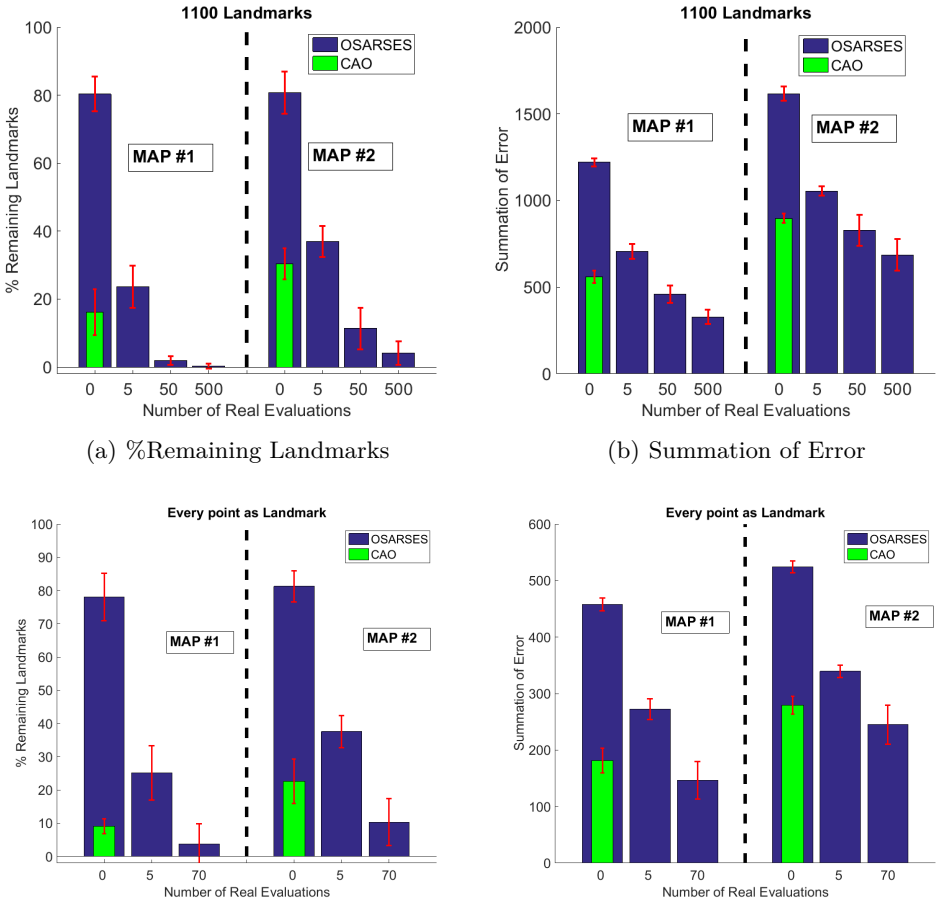
(c) %Remaining Landmarks



(d) Summation of Error

Figure 4.5. *Conducted Experiments for 3 AUVs, for both the algorithms with 1100 landmarks and every point as landmark*

`CAOnavigation`.



(a) %Remaining Landmarks

(b) Summation of Error

(c) %Remaining Landmarks

(d) Summation of Error

Figure 4.6. *Conducted Experiments for 10 AUVs, for both the algorithms with 1100 landmarks and every point as landmark*

The results indicate clearly that the CAO approach, can obtain better performance even from the OSARSES algorithm which uses five real positions before being able to produce its control decision. Regarding to the 3 robots experiment (figure 4.5), an improvement of about 79.7% and 24.1% in MAP #1 as well as 147.3% and 9% in MAP #2 map is achieved, compared to the OSARSES with 0 and with 5 candidate set of waypoints ($\mathbf{X}^{R,cand}$) respectively. The study of the summation of error, has further strengthened the conclusions that the proposed approach outperforms the performance of the OSARSES in cases of zero and 5 candidate set of waypoints. Inevitably, in case where the OSARSES uses

a large number of candidate set of real waypoints, it performs better than CAO. As it is seen in figure 4.6, in case of 10 robots with 1100 landmarks, the proposed algorithm can scale up well, retaining its improvements' levels. Concretely, it achieves an improvement of about 80.2% and 71.4% in each map respectively (Figure 4.6(a)), with a corresponding improvement in the summation of error (Figure 4.6(b)). Interestingly, in the scenario where every point of the map is considered as landmark and as a result a fine-grained mapping is required, the proposed algorithm is able not only to outperform the 0 and 5 real evaluation cases of OSARSES (acquiring the impressive improvement of 88.4% and 63.8% in Map #1 as well as 72.3% and 40% in MAP #2 respectively), but also to approximate the performance of OSARSES with 70 real evaluation per timestep.

The aforementioned result is not out of the blue. As the number of robots and the landmarks is increasing, the 70 real evaluation became a rather insufficient number for the OSARSES algorithm. Unfortunately, the further increasing is prohibited even in the simulation test-bed and it will take forever to statistically remove the randomness of the results. On the contrary, in the case of CAO approach we are able to efficiently/securely increase the number of candidates, due to the fact that

- these candidates are not actually evaluated, so the operational cost is zero

- even the computational cost to test the candidates on the CAO's estimator (equation 4.20) is extremely inferior compared to the one of OSARSES which has to calculate all the terms of equation 4.9 for every single candidate configuration.

4.6 Experimental Results

The proposed methodology has been also evaluated through real-life tests concerning sea-floor mapping of unknown areas using two AUVs (**A**utonomous **U**nderwater **V**ehicles). The tests were conducted in the Leixes Por, located in the city of Oporto, Portugal[7]. Next we describe the details of the experiments.

[7] A video demonstrating the real-life experiments can be found here: `https://tinyurl.com/NoptilusMapping`

4.6.1 System Details

A schematic diagram of the system developed for implementing the proposed approach is illustrated in Figure 4.7.

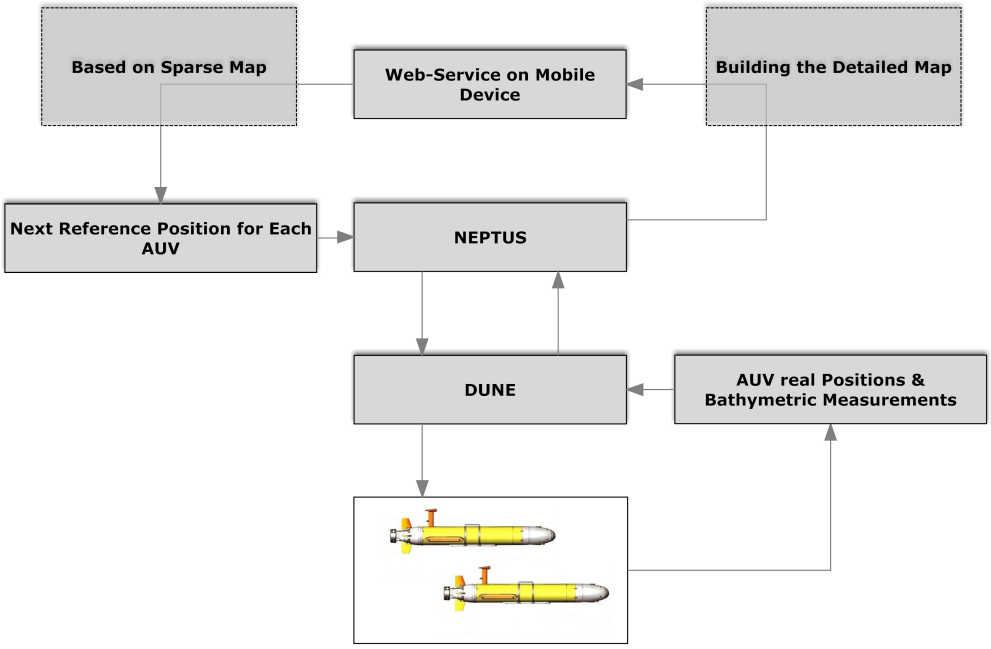


Figure 4.7. *Flowchart of system used in the experiments*

The entire procedure can be separated into 2 parts: a web-service one and the client-side part. The web-based interface undertakes the role of coordinating the AUVs. This service requires from the operators, minimum amount of information such as the number of the vehicles, so as to begin the procedure. The web-application was designed to provide the operators with a general-purpose tool through which the progress of mission is streamlined in real-time. It is worth noticing that the web-service is compatible with the existing systems and can be adopted so as to navigate any type of AUV. The web-service produces in every time step the next optimal position (waypoint) for each AUV, based on the proposed approach, the landmarks estimates, and the current position of the AUVs. Next, the new waypoints are transferred to the client-side part of the application.

The client-side part consists of 2 software: NEPTUS[8] and DUNE[9]. NEPTUS is a command and control software that can be used to plan,

[8]Neptus Command and Control Software: http://www.lsts.pt/toolchain/neptus/
[9]DUNE: Unified Navigation Environment: http://www.lsts.pt/toolchain/dune

simulate, monitor and review missions executed by AUVs. DUNE is the runtime environment for vehicle on-board software. It is used to write generic embedded software at the heart of the vehicle, e.g. code for control, navigation, communication, sensor and actuator access, etc. Through these tools, AUVs receive the information about the next waypoint.

When the AUVs reach their respective desired location and stabilize, they activate their exteroceptive sensors. In the sequel, the AUVs communicate their sensor measurements together with their exact positions through NEPTUS and DUNE to the web-service.

Subsequently, the web-service incorporates the actual positions of the AUVs into the proposed model as well as updates the landmarks estimates, based on the sensor information received. A non-linear least squares algorithm is used perform such an estimation (SLAM) procedure. At the same time the web-service, based on the sensor measurements and by employing a Gaussian based interpolation algorithm, builds the detailed map. It has to be emphasized here that the overall procedure fully relies on each AUV "local" localization system (i.e., the localization system of each AUV does not incorporate measurements from the other AUVs) and, thus, the SLAM system only performs landmark estimation.

4.6.2 Ground Truth - Usual Practice

Through earlier measurements, observations and calculations, a detailed map of each region we want to capture is available. Hereafter, we will refer to this version of the map as *ground truth*. To obtain a ground truth map, several AUVs have to operate –in a non-cooperative fashion– for many hours using multi-beam sensors, following a predefined iterative procedure, collecting an enormous amount of measurements. Apparently, this is a very time consuming and expensive task. The *ground truth* map, mainly due to the fact that it is considered the best available perception about the morphology of the seabed, constitutes the reference map, the one that is going to be used in order to evaluate, in terms of accuracy, the exploration's results of alternative, feasible methodologies

Simultaneously, several *usual practice* versions of each map are also available. These maps were captured using the today's usual practice for the exploration of unknown underwater environments. According to this approach, the AUVs are following a predefined trajectory collecting data from the seabed using a specific sampling rate, until a predefine time. This methodology despite its simplicity, holds many advantages, that have established its usage in the most real-life exploration/coverage missions:

- predictability (a priori information about the morphology of the area can be easily incorporated in the global plan by designer).
- it does not require any online communication link -the paths are predefined, and the measurements are gathered at the completion of the experiment - between them or ground/vessel station, minimizing the energy consumption.
- fully-coverage is guaranteed.
- The navigation-scheme (straight lines) enhances the localization accuracy.

On the other hand, the *usual practice* has some vital drawbacks that limit its performance:

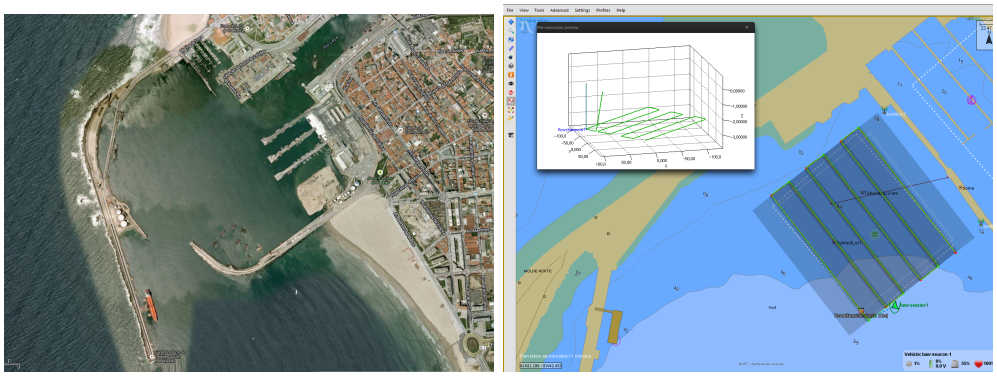
- The navigation scheme is constant, dealing the same way (number of samples), sub-areas with different height discrepancies or in more abstract terms with different kind of importance.
- For realistic time/energy consuming missions, there is always a risk of completely missing some important sub-part of the area, which is located between the AUV's paths.
- The coordination/cooperation of more than one AUV is not trivial and in the most cases does not taking into consideration the initial positions of the AUVs.
- Human intervention is necessary to appropriately define the multi-robots' not-overlapping routes.

It is emphasized that the proposed approach does not use any information from the ground truth map or the usual practice map. Both these two maps (*usual practice* and *ground truth*) are needed for evaluation purposes.

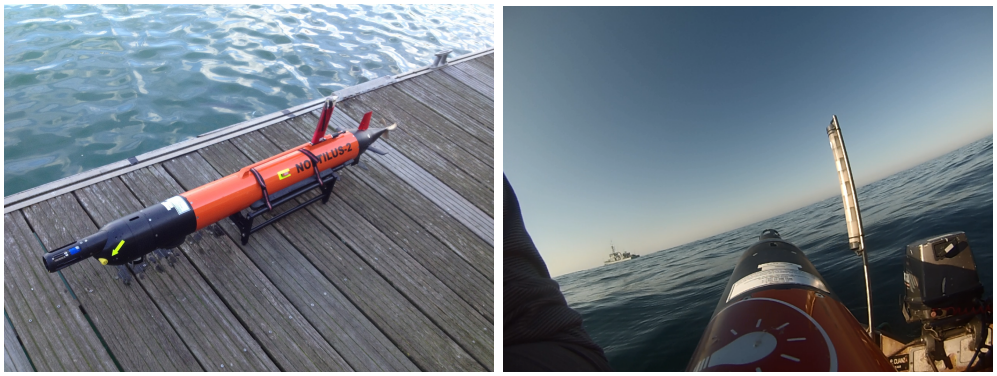
4.6.3 Experiments in Oporto's Harbor

The objective of the experiments is to build a detailed map of two different sub-areas of the Oporto's harbor (Figure 4.8(a)). To accomplish such a mission, in each case, 2 AUVs (Figure 4.8(c)) are deployed equipped with single beam bathymeter sensor. It's worth mentioning that such sensors provide us a small amount of information about the sub-region where the AUVs are deployed, practically only one point.

The experimental environment can be described as follows:



(a) Deploying area, Oporto harbor (b) Neptus - Command and Control Software

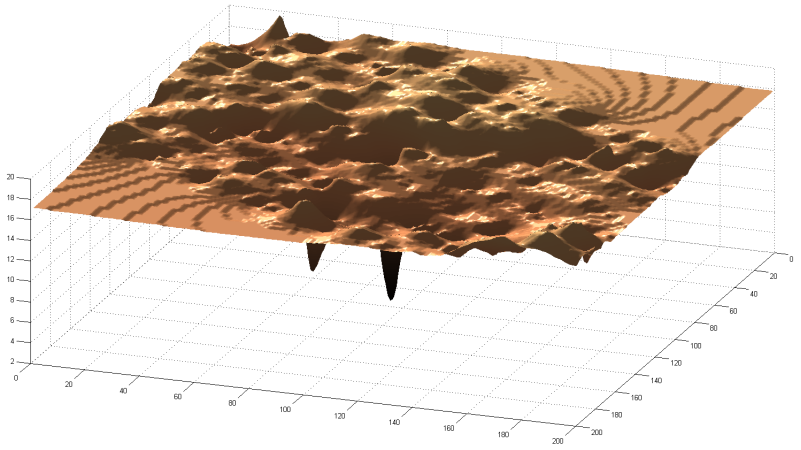


(c) Autonomous Underwater Vehicle (d) AUV in operation mode

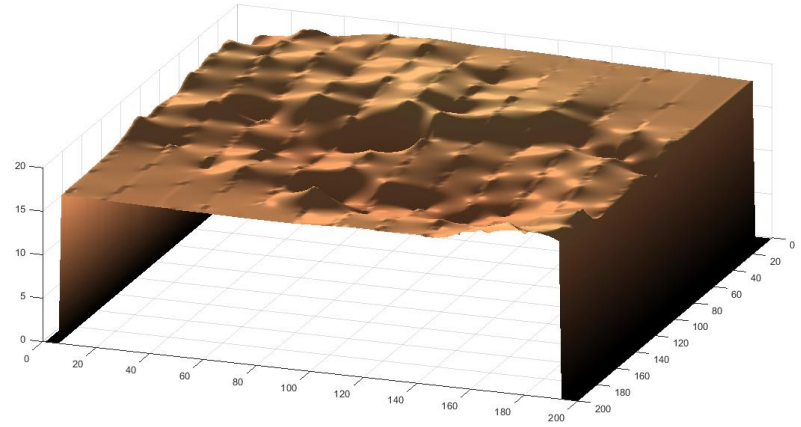
Figure 4.8. *The available modules(hardware/software) for real-word Experiment*

- In both cases the map is a square area with dimensions equal to 100×100 meters.
- The number of AUVs is equal to $N_R = 2$.
- The AUVs are moving within the terrain's limits, i.e., within $[X_{min}, X_{max}]$ and $[Y_{min}, Y_{max}]$ in the x- and y-axes, respectively. Each AUV remain in constant z so as to neglect any collision possibility.
- Experiment contacted for $T = 500$ time-steps.

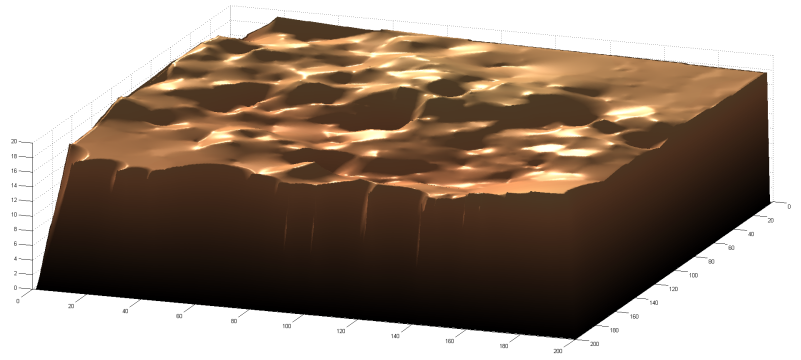
In each time step, the web-service navigates the AUVs to the next desired position through the NEPTUS (Figure 4.8(b)) application and builds the detailed map. Overall, upon completions of the procedure, 500×2 single beam bathymeter sensor measurements are available, in



(a) *Ground Truth* Map

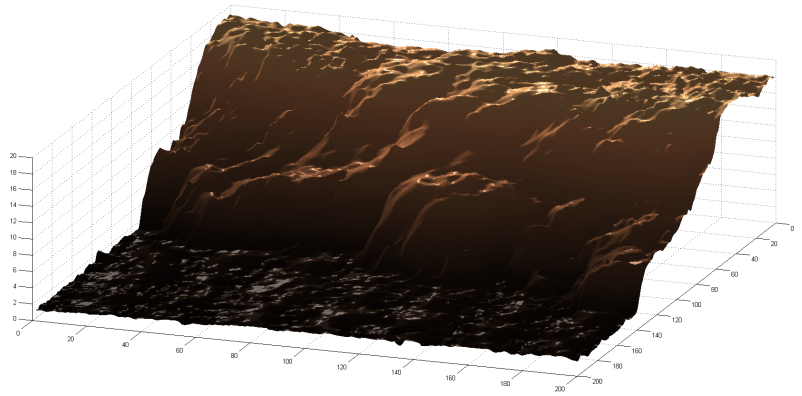


(b) *Usual Practice* Map

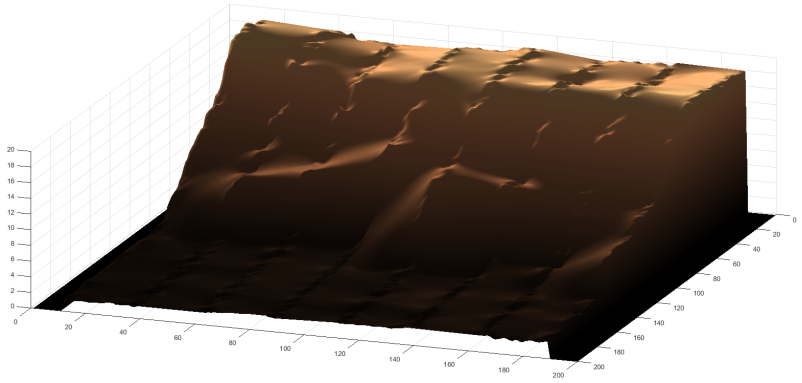


(c) *Proposed Approach* Map based on 1000 Samples

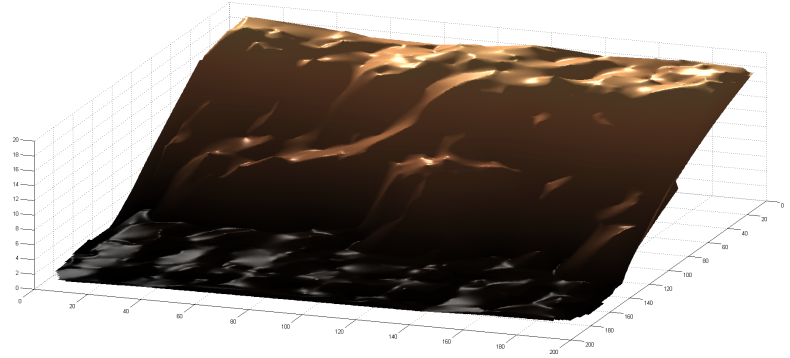
Figure 4.9. Ground truth, Usual Practice *and produced by the* Proposed Approach *Map Employing 2 AUVs for #Sharp_Surface Map*



(a) *Ground Truth* Map



(b) *Usual Practice* Map



(c) *Proposed Approach* Map based on 1000 Samples

Figure 4.10. Ground truth, Usual Practice *and produced by the* Proposed Approach *Map Employing 2 AUVs for #Slop_Surface*

order to form the high detailed version of the observable area. Figures 4.9 and 4.10 illustrate the *ground truth*, the *usual practice*, as well as the generated, from the *Proposed CAO-based Approach*, version of the maps for the 2 areas on with we performed the experiments respectively. The first map, will be referred hereafter as #Sharp_Surface while the second one as #Slop_Surface[10].

In order to evaluate the effectiveness of the proposed approach, we calculate the $\mathsf{L}^2-\mathsf{Norm}$[11], between the *ground truth* and the *usual practice* map as well as between the *ground truth* and the *Proposed Approach* map. Alongside the accuracy of the *Proposed Approach* map, we also evaluate the reliance of the proposed approach on the number of measurements. The results are depicted in Tables 4.2 and Table 4.3. Apart from the estimation-accuracy, expressed in terms of Euclidean norm, we display the number of points (measurements) that were used, to present an overview about the total work that is needed in each case.

Approach	Number of Samples	% of samples, w.r.t. ground truth (27961 samples)	L2-norm accuracy
Proposed	**1000**	**3.58%**	**609.88**
Usual	7503	26.83 %	900.27

Table 4.2. $\mathsf{L}^2-\mathsf{Norm}$ *and Number of Samples, between the* ground truth *version of the #Sharp_Surface map vs the* Proposed Approach*(cao-generated) map and the* Usual Practice *map*

As one can see, the proposed approach, using only 1000 measurements (86.7% reduction in the #Sharp_Surface and 77.8% in the #Slop_Surface), is able to construct a map with more than 32% accuracy in the first case and 2.1% in the second, as compared to the today's usual practice.

[10]Please note that both interpolated versions of *usual practice* present some ridges along the constructed terrain. These ridges correspond to the areas where the AUVs traversed and therefore the samples' concentration is greater than the rest of the terrain.

[11]In order to implement this, at first we discretize, with a sufficient small step, the areas to be compared and afterwords we apply the $\mathsf{L}^2-\mathsf{Norm}$ on the vectorized versions of the sampled areas.

Approach	Number of Samples	% of samples, w.r.t. ground truth (35846 samples)	L2-norm accuracy
Proposed	**1000**	**2.79%**	**828.17**
Usual	4509	12.58 %	846.24

Table 4.3. $\mathrm{L}^2 - \mathrm{Norm}$ *and Number of Samples, between the* ground truth *version of the #Slop_Surface map vs the* Proposed Approach*(cao-generated) map and the* Usual Practice *map*

4.7 Conclusions

A novel method for dealing the problem of exploring an unknown area using multi-robot teams under environmental and communication constraints, while simultaneously building a detailed map of the environment has been proposed. Based on this approach we are transforming a standard trajectory generation problem so as to optimize a transformed version of trajectory generation efficiency, employing CAO algorithm. The methodology proposed is independent of requirements regarding operational characteristics of the robots like communication range and type of sensors.

Additionally, the proposed scheme, in relation to the vast majority of the optimal/dynamic programming approaches, takes into account the non-linear characteristics of the robots' sensors and the fact that the operation area is unknown. In a nutshell, the proposed methodology aims to bridge the gap between the state-of-the-art algorithms and the actual practices, by successfully navigating the robots through environments, where the objective/reward function cannot be calculated a-priori, due to the afore-mentioned reasons.

The applicability and adaptability of our approach in realistic scenarios has been demonstrated through simulated and real-life underwater sea-floor mapping experiments in the port of Porto, Portugal using a team of AUVs. The proposed approach is independent of the SLAM methodology employed since it is based on the approach "to do the best it can be done" based on the current configuration, given the communication/sensing/SLAM system, allowing even cases where the multi-robot team comprises of vehicles with mutually different sensing capabilities or operating different SLAM algorithms.

With an outlook to the future work, we consider incorporating the

localization problem to our decision-making mechanism. The proposed approach can be safely classified under the spectrum of the optimization based ones. These algorithms allow to interface additionally/secondary objectives, by appropriately modifying/revising the performance criterion. Moreover, we will focus our efforts on the development of an approach that will primarily retain all the achievements of the proposed method (operate under unknown terrain, without actual evaluate the candidate configurations, etc.), but at the same time will able to provide near-globally-optimal solutions for all the experiment's horizon and not only for the next timestep. In order to build such a non-greedy algorithm, we should apply an offline learning scheme where the algorithm will translate the sensors' measurements into new commands'/robots' directions, by applying some transformation on them, that has been learnt from numerous simulation or/end real-life experiments.

5 Distribute Online Multi-Robot Model-free Approach

Contents

5.1 Introduction

In the final chapter of this thesis, we deal with multi-robot tasks where the user-defined objectives of the mission can be casted as a general optimization problem, without explicit guidelines of the sub-tasks per different robot. Due to the unknown environment, unknown robots' dynamics, sensor nonlinearities, etc., the analytic form of the optimization cost function is not available a priori. Therefore, standard gradient descent-like algorithms are not applicable to these problems[1]. To tackle this, we introduce new resource optimization algorithm – specifically tailored to the context of multi-robot applications – that extends the Cognitive-based Adaptive Optimization (CAO) algorithm (as proposed in the previous chapter).

In a nutshell, an update cycle on decision variables of the proposed algorithm consists of the following steps. Initially, the robots' measurements are gathered in a central node (robot-leader or base station) where the calculation of the global objective function takes place. In the sequel, each robot's contribution to the cost function is approximated and forwarded to the corresponding robot. In a fully distributed fashion, each robot constructs a linear-in-the-parameters estimator to approximate the (unknown - problem dependent) evolution of its sub-cost function. Then, each robot generates random (or pseudo-random) perturbations around its current state and neglects the ones that violate the operational constraints (if any). Finally, the next robot's action is the one valid perturbation that achieves the best score on the previously constructed estimator.

The primary deviation of the proposed approach, compared to the original version of CAO, is in its distributed nature. More precisely, although each robot does not know explicitly either the decision variables of the other robots nor their measurements of them, it is able to update its own decision variables effectively, in a way to cooperatively achieve the team objectives. The later can be achieved through an exclusive for each robot cost function, designed so as to encapsulate not only the mission objectives but also the other robots' dynamics (for more details see section 5.3). Rigorous arguments establish that, despite the fact that the dynamics that govern the multi-robot system are unknown, the proposed methodology shares the same convergence characteristics as those of block coordinate descent algorithms [68]. As it is exhibited in the presented applications, the distributed nature of the proposed algorithm allows, also, the rapidly convergence, especially in cases with many robots, each of

[1]The reader is referred to section 2.2 for an succinct discussion regarding the classes of alternative methodologies, presenting their advantages and disadvantages.

which with several decision variables.

The contributions with respect to the multi-robot approaches, as presented in chapter 2, are the following:

(i) The problem is formulated in a continuous domain without the need to either know all the states and measurements beforehand, or to perform a relaxation on the original multi-robot problem (*optimal control & dynamic programming* approaches). The ability to cope with unknown dynamics (robots-environment) and unknown cost functions, imparts a generality to the proposed algorithm, regarding the spectrum of applications that can be utilized.

(ii) However, the main advantage of the proposed algorithm is that it does not require, either a priori calculation of the cost function (*optimal-one-step-ahead* approaches) or the analytical form of the system to be optimized to be explicitly known (*optimal control & dynamic programming* approaches). Instead, the proposed algorithm can cope with cost functions, the calculation of which can only be achieved by actual performing the corresponding course of actions. In the same vein, the proposed algorithm does not need to actually evaluate the decisions variables in the vicinity of their current values, so as to appropriately calculate the corresponding update on them. The proposed algorithm instead, is able to find the (locally) optimal configuration for the decision variables, by using only noise-corrupted measurements collected from the robots' sensors.

(iii) Furthermore, instead of relying on exhaustive, computational-intensive simulations (*simulation-based* approaches), the proposed scheme is able to on-line learn the problem specific characteristics that affect the user-defined objectives. By doing so, the proposed algorithm does not need any elaborate model, in order to learn its decision-making mechanism.

It has to be emphasized that apart from rendering the optimization problem practically solvable, the proposed approach preserves additional features that make it particularly tractable:

(i) its complexity is low, allowing *real-time implementations*;

(ii) it can handle a variety of *physical constraints*;

(iii) it has *fault tolerant characteristics*, i.e. on-line redesign in cases of: one or more robots are added or removed, an extra task has been added to the set of objectives, etc.;

(iv) it is able to adapt its behavior even in cases where a *time-varying*

objective function is employed[2].

The proposed control strategy is evaluated on *four different simulation set-ups* under multiple scenarios, against both general purpose and specifically-tailored to the problem in hand, algorithms. All the simulation set-ups have been chosen so as i) the objective of the multi-robot mission can be expressed in cost function ii) the evaluation of which cannot be performed beforehand. In the first simulation set-up, the objective is to spread out the robots over a 2D environment, while aggregating in areas of high sensory interest. What is important about this set-up is the fact that, the robots do not know beforehand where the areas of sensory interest are, but they learn this information on-line, utilizing sensor measurements from their current positions. The proposed algorithm is evaluated together with the approach as proposed in [1] for the problem in hand. In the second simulation set-up, the trajectories of the robots should be designed in real-time having a twofold objective (which forms a trade-off). On one hand, the part of the 3D terrain that is monitored (i.e. visible) by the robots have to be maximized and on the other hand, for each one of these visible points in the terrain, the closest robot has to be as close as possible to that point. This problem along with a centralized CAO-based methodology has been proposed in [2], therefore a detailed analysis regarding the performance of both algorithms, in different scenarios, is presented. Moreover, the proposed methodology is evaluated in a time-varying formation control scenario for a group of robots. In this simulation set-up, the robots have to perform a user-defined trajectory, while they are holding a pre-specified formation [105]. Last but not least, the proposed methodology is evaluated in the task of persistent coverage. The objective of this application is to maintain a user-defined level of coverage in an unknown environment [49]. This is a quite challenging task as the mission objectives constantly change, while the unknown morphology of the environment does not allow the prior calculation of the improvement in coverage task. Conclusively, if it is possible to define a cost function which encapsulates the mission objectives and can be calculated through the robots' measurements for every decision variables configuration, the proposed methodology will be directly applicable to the corresponding problem.

The remainder of the chapter is structured as follows: section 5.2 presents the translation of a general purpose multi-robot framework to a constrained optimization problem, highlighting the difficulties and the

[2]Please note that the rate of change in the objective function should be smaller than the learning capabilities of the algorithm (see section 5.2)

obstacles of the general problem. The description of the proposed algorithm, that tackles such a problem, is presented in section 5.3. sections 5.4, 5.5 and 5.7 present four indicative multi-robot applications: *adaptive coverage of unknown 2D environment*, *3D surveillance of unmapped terrains*, *time-varying formation control* and *persistent coverage of unknown 2D environments*, respectively. In all these sections, we perform a series of simulations in different scenarios to adequately analyze the performance of the proposed algorithm. The overall conclusions of the chapter are drawn in section 5.8.

5.2 Problem formulation

Consider a team (swarm) which consists of N robots interacting with each other, towards achieving a global set of objectives. Let's assume the following augmented decision vector

$$\mathbf{x}(k) \equiv [x_1^\tau(k), x_2^\tau(k), \ldots, x_N^\tau(k)]^\tau \tag{5.1}$$

where $x_i(k) \in \mathbb{R}^n$ denotes the decision variables of the i-th robot at k iteration. These decision variables represent the controllable parameters of the available robots (e.g. position, motors, propellers, thrusters, rotation of the cameras, etc.). Furthermore, the augmented vector which contains the available exteroceptive measurements takes the form

$$\mathbf{y}(k) \equiv [y_1^\tau(k), y_2^\tau(k), \ldots, y_N^\tau(k)]^\tau \tag{5.2}$$

where $y_i(k) \in \mathbb{R}^m$ denotes the measurement vector of the i-th robot at k iteration and its evolution can be represented as

$$y_i(k) \equiv h_i(k, x_i(k)) \tag{5.3}$$

where $h_i(\cdot)$ denotes an unknown, nonlinear function that depends on both $x_i(k)$ and the specific problem characteristics.

The accomplishment of the multi-robot system's objectives (e.g. mapping, surveillance, coverage, etc) can be translated to the minimization (or maximization)[3] of a specifically defined global cost function J_k, i.e.,

$$J_k \equiv \mathcal{J}\Big(x_1(k), x_2(k), \ldots, x_N(k)\Big) \tag{5.4}$$

where $\mathcal{J}(\cdot)$ is a non-negative, nonlinear, scalar function which depends – apart from the robots decision variables – on the particular dynamics of the

[3]Without loss of generality, in the rest of the chapter we assume a minimization problem.

problem (e.g. the environment where the robots live and interact). Due to the dependence of the function $\mathcal{J}$ on the particular problem characteristics, the *explicit form of the function $\mathcal{J}$ is not known* in practical scenarios; as a result, standard optimization algorithms (e.g. gradient descent) are not applicable. However, in most practical cases, the current value of the objective function can be approximated from the robots' measurements,

$$\mathcal{J}\Big(x_1(k),\ldots,x_N(k)\Big) = J\Big(y_1(k),\ldots,y_N(k)\Big) + \xi_k \tag{5.5}$$

where ξ_k denotes the noise introduced in the estimation of $\jmath_k$, due to the presence of noise in the robots sensors.[4] It must be emphasized that, contrary to $\mathcal{J}$, J can be evaluated "offline", if the measurement vector $\mathbf{y}(k)$ is available. However, the *acquisition of a new measurement vector* requires an *actual evaluation* of the decision variables on the robotic system (5.2),(5.3).

Apart from the problem of dealing with a criterion for which, an explicit form in not known but only its noisy measurements are available at each time, the decision vector $\mathbf{x}(k)$ should satisfy a set of constraints that, in general, can be represented as follows:

$$\mathcal{C}\left(\mathbf{x}(k)\right) \leqslant 0 \tag{5.6}$$

where $\mathcal{C}$ is a set of non-linear functions of the decision variables $\mathbf{x}(k)$. As in the case of $\mathcal{J}$, the constraints function $\mathcal{C}$ depends on the particular problem characteristics and an explicit form of this function may be not known in many practical set-ups; however, it is natural to assume that the low-level algorithm is provided with information whether a particular selection of decision variables $\mathbf{x}(k)$ satisfies or violates the set of constraints (5.6).

Given the mathematical description presented above, the problem of on-line choosing the decision variables for a multi-robot system, so as to accomplish a set of objectives, can be mathematically described as the following constrained optimization problem:

$$\begin{aligned} &\text{minimize} && \jmath_k \\ &\text{subject to} && \mathcal{C}\left(\mathbf{x}(k)\right) \leqslant 0 \end{aligned} \tag{5.7}$$

[4] Please note that, although it is natural to assume that the noise sequence ξ_k, is a stochastic zero-mean signal, it is not realistic to assume that it satisfies the typical Additive White Noise Gaussian (AWNG) property, even if the robots sensors do; as $\mathcal{J}$ is a nonlinear function of the robots decision variables and thus of the robots sensor measurements (5.3), the AWNG property is typically lost.

As already noticed, the difficulty in solving, in real-time the constrained optimization problem (5.7) lies in the fact that explicit forms for the function $\mathcal{J}$ and $\mathcal{C}$ are not available. Although this is not the only problem, jointly optimizing a function over multiple robots (N), each of which with multiple decision variables (n) can incur excessively high computational cost.

5.3 Proposed algorithm

Having defined the fundamental aspects that govern a multi-robot application, we will proceed on presenting the proposed algorithm for updating the decision variables $\mathbf{x}(k)$ so as to minimize the cost function (5.4) subject to (5.6).

(*STEP 1*) As first step, and for each iteration k, the robots transmit the acquired measurements, after the execution of $\mathbf{x}(k)$ decision variables.

It must be emphasized that, this step can be performed even in cases where global communication between all robots is not feasible. In such case, each robot can send and receive measurements to and from peer (adjacent) robots, until all the measurements aggregate to the corresponding processor unit (robot or ground station). The latter can be guaranteed by introducing an extra condition on the constraints set (5.6), ensuring the connectivity – if applicable to the problem in hand – among the different robots (e.g. the *communication range* constraint in chapter 4).

(*STEP 2*) Thus, the global cost function can be straightforwardly derived from [see (5.4) and (5.5)]:

$$\jmath_k = J\Big(y_1(k), \ldots, y_N(k)\Big)$$

Additionally, for each i-th robot calculate the following discrepancy:

$$\begin{aligned} &\Delta_i(k) \equiv \jmath_k - \\ &J\Big(y_1(k), \ldots, y_{i-1}(k), y_i(k-1), y_{i+1}(k) \ldots, y_N(k)\Big) \end{aligned} \tag{5.8}$$

In other words, $\Delta_i(k)$ encapsulates the effect of the $x_i(k)$ on the current problem for the k timestamp.

Please note that, since

$$J\Big(y_1(k), \ldots, y_{i-1}(k), y_i(k-1), y_{i+1}(k) \ldots, y_N(k)\Big)$$

is analytically available, we can calculate this term, although the resulting value does not necessary correspond to the actual value when the robots have the following decision variables:

$$\{x_1(k), \ldots, x_{i-1}(k), x_i(k-1), x_{i+1}(k) \ldots, x_N(k)\}$$

Although, there may be a discrepancy between the way we calculate $J(\cdot)$ and its actual value, that does not affect the convergence properties of the proposed algorithm. This discrepancy is application oriented and depicts the effect of other robots' decisions on each robot's measurements. If the measurements acquired from a robot only affects its own decision variables and the problem itself (5.3), then there is no discrepancy at all.

(*STEP 3*) Next, the calculated discrepancy $\Delta_i(k)$ is sent to the i-th robot. After this step all the calculations are performed locally, building a system that is resilient to robot failures, does not require any global coordination and all the decision variables' updates are made in a (parallel) distributed fashion.

(*STEP 4*) Each i-th robot – at the same k-th iteration – performs the following:

• Calculate the $J_i(k)$ that corresponds to the last executed decision variables $x_i(k)$ as:

$$J_i(k) = J_i(k-1) + \Delta_i(k),\ \forall k \geqslant 1,\ J_i(0) = \jmath_0 \tag{5.9}$$

Therefore, each robot is responsible to choose the next values for its decision variables $x_i(k+1)$, having as only objective the minimization of its corresponding cost function $J_i(\cdot)$[5]. Each such sub-problem is a lower-dimensional minimization problem, and thus can typically be solved more easily than the full problem.

• Construct an estimator for $J_i(k+1)$:

$$J_i(k+1) \approx \hat{J}_i(k+1) = \theta_i^\tau(k)\phi_i\big(x_i(k)\big) \tag{5.10}$$

where ϕ_i denotes the non-linear vector of L *regressor terms*, θ_i denotes the vector of the *parameter estimates* and L is a positive user-defined integer which denotes the size of the function approximator (5.10). The vector ϕ_i of regressor terms must be chosen so that it satisfies the so-called *Universal Approximation Property* [106], i.e. it must be chosen so that the approximation accuracy of the constructed approximator (5.10) is an increasing function of the approximator's size L. Polynomial approximators, radial basis functions, kernel-based approximators, etc. are known to satisfy such a property [106].

[5] In general case: $\sum_{i=1}^{N} J_i(k) \neq \jmath_k$ and $\prod_{i=1}^{N} J_i(k) \neq \jmath_k$, $\forall k$

Algorithm 3 ϕ_i construction

Require: maxorder, $L_1, L_2, \ldots, L_{\text{maxorder}}$, x_i, n
Ensure: ϕ_i

```
φ_i = 1
for j ∈ {1, ..., maxorder} do
    for ν ∈ {1, ..., L_j} do
        g = 1
        for l ∈ {1, ..., j} do
            Generate r := random integer ∈ {1, ..., n}
            g = g · x_i^(r)
        end for
        φ_i = [φ_i^τ, g]^τ
    end for
end for
```

Experimenting with different types of ϕ_i, in different multi-robot set-ups (sections 5.4-5.7 and [55, 56]), it was found that it is sufficient to construct a polynomial estimator as in algorithm 3. The tunable parameters of this procedure are the maximum order of monomials (maxorder) and the corresponding number of monomials per order $L_1, L_2, \ldots, L_{\text{maxorder}}$, where $L_1 + L_2 + \cdots + L_{\text{maxorder}} = L - 1$ should be hold. Mathematically speaking, the number of different monomials per order is given by the number of possible combinations with repetitions (multiset coefficient):

$$\left(\!\binom{n}{i}\!\right) = \binom{n+i-1}{i} = \frac{n(n+1)(n+2)\cdots(n+i-1)}{i!}$$

where $\binom{a}{b}$ denotes the binomial coefficient. However, the summation $L_1 + L_2 + \cdots + L_{\text{maxoder}}$ may exceed the number of available monomials $L-1$. A usual practice is to downscale the number of monomials as follows:

$$L_i = \left[\binom{n+i-1}{i} s\right] \tag{5.11}$$

where $[\cdot]$ denotes the nearest integer and s denotes the following scaling factor:

$$s = \frac{L-1}{\sum_{i=1}^{\text{maxorder}} \binom{n+i-1}{i}}$$

Having defined the vector of regressor terms ϕ_i, the estimator vector θ_i can be calculated using standard least-squares estimator principles, i.e., θ_i is obtained by solving the following optimization problem:

$$\theta_i(k) = \underset{\vartheta}{\operatorname{argmin}} \sum_{\ell=k-T(k)}^{k-1} \left(\vartheta^\tau \phi_i\big(x_i(\ell)\big) - J_i(\ell+1) \right)^2 \tag{5.12}$$

where $T(k)$ denotes the time-window over which the least-squares estimation is taking place. There are several algorithms for solving the least-square problem (normal equation, QR decomposition, SVD, etc.). Although, singular value decomposition (SVD) is more computational intensive in comparison to other alternatives, we utilize this approach due to the fact that it is more numerical stable (e.g. when the problem is ill-conditioned) [107].

• Choose a positive function $\alpha(k)$ to be either a constant positive function or a time-descending function satisfying $\alpha(k) > 0, \sum_{k=0}^{\infty} \alpha(k) = \infty, \sum_{k=0}^{\infty} \alpha(k)^2 < \infty$. Although, the proof of convergence (as presented in the next sub-section) can be established for both the choices, in our simulation experiments, we found out that the choice of a constant positive is more robust.

• Generate (randomly or pseudo-randomly) a set of M valid candidate perturbations:

$$\delta x_i^{(1)}(k), \delta x_i^{(2)}(k), \ldots, \delta x_i^{(M)}(k)$$

where $\delta x_i^{(j)}(k)$ are vectors of the same dimension as $x_i(k)$ and M is a positive integer that is larger[6] than 2n. A candidate perturbation j is considered valid if[7] :

$$\mathcal{C}\left(\left[x_1^\tau, \ldots, x_{i-1}^\tau, x_i^\tau + \delta x_i^{(j)}, x_{i+1}^\tau, \ldots, x_N^\tau \right]^\tau \right) \leqslant 0 \tag{5.13}$$

The random choice for the candidates is essential and crucial for the efficiency of the algorithm, as such a choice guarantees that $\hat{J}_i(k+1)$ is a reliable and accurate estimate for $J_i(k+1)$; see [90] and [91] for more details.

• Estimate the effect of each of the candidate perturbations to the current vector $x_i(k)$ by employing the previously constructed estimator (5.10) and pick the candidate perturbation with the "best" effect, i.e., choose the vector $\delta x_i^{(j^*)}(k)$ that satisfies

$$\delta x_i^{(j^*)}(k) = \underset{j=1,\ldots,M}{\operatorname{argmin}} \theta_i^\tau(k) \phi_i\left(x_i(k) + \alpha(k) \delta x_i^{(j)}(k) \right)$$

[6] See [90] for more details about the sufficiency of this condition.

[7] The distributed nature of the algorithm may impose a stricter set of constraints, in comparison with cases where a centralized control is applied.

• Update the i-th robot decision variables as:

$$x_i(k+1) = x_i(k) + \alpha(k)\delta x_i^{(j)}(k) \tag{5.14}$$

Remark 1 The above distributed update of the decision variables (*STEP 4*) does not need information about what is happening to the rest of the robots. All the needed information has been "packed" to the scalar value $\Delta_i(k)$. At each iteration, each robot attempts to minimize the objective function $J_i(k)$ by assuming that the other robots' decision variables are part of the problem to be solved.

5.3.1 Convergence analysis

Remark 2 As shown in [90, 91], the distributed algorithm implemented in each robot (*STEP 4*) guarantees that: If $M \geqslant 2 \times \dim(x_i)$, the vector ϕ satisfy the Universal Approximation Property and the functions J_i and C are either continuous or discontinuous with a finite number of discontinuities, then the update rule of x_i (5.14) is equivalent with:

$$x_i(k+1) = x_i(k) - A(k)\nabla_{x_i} J_i + \epsilon(k)$$

where $A(k)$ is a positive definite matrix which depends on the choice of α and $\nabla_{x_i} J_i$ denotes the gradient of J_i with respect to the x_i decision variables. Additionally, $\epsilon(k)$ is a term that converges exponentially fast to the subset $\mathcal{D} = \{\epsilon(k) : |\epsilon(k)| \epsilon\}$, where ϵ is a positive constant that can be made arbitrarily small [by increasing the size L of the estimator (5.10)].

The next theorem describes the properties of the proposed methodology; as the proof of this theorem is along the same lines as in [108, Proposition 2.7.1], only a sketch of proof in provided.

Theorem 3 *The local convergence of the proposed algorithm can be guaranteed in the general case where the global cost function $\mathcal{J}$ and each robot's contribution J_i are non-convex, non-smooth functions.*[8]

Sketch of the proof: By using Remark 2 (*projected gradient-descent* on the minimization of J_i) and equations (5.8)-(5.9), we can establish that the distributed update on each robot is equivalent with:

$$x_i = \underset{w}{\operatorname{argmin}}\, \mathcal{J}(x_1, \ldots, x_{i-1}, w, x_{i+1}, \ldots, x_N)$$

[8] Moreover, recent studies imply that BCD methodologies can achieve global convergence even in cases where the global cost function (5.4) is non-convex but holds some properties. For example, in [109] the authors establish global convergence of the BCD algorithm in the general case where the global cost function $\mathcal{J}$ and each robot's contribution J_i are non-convex functions, but the so-called Kurdyka-Łojasiewicz (KL) property is satisfied.

subject to (5.13), and therefore, the proposed algorithm approximates the behavior of the Block Coordinate Descent (BCD) [68, Algorithm 1] family of approaches. Following the proof described in [108, Proposition 2.7.1], it is straightforwardly to see that: if the minimum with respect to each block of variables is unique, then any accumulation point of the sequence $\{x(k)\}$ generated by the a BCD methodology is also a stationary point.

5.3.2 Complexity

For *STEPS 1-3* the computational burden is accumulated in the calculation of $\Delta_i(k)$ (5.8) for each i robot. However, the calculation of $J(\cdot)$ is problem-dependent, thus, it is not possible to analytically derive bounds regarding its complexity. In the reported cases [cost functions (5.19),(5.21),(5.23),(5.26) and (5.30)], as well as in the most real-world applications, $J(\cdot)$, computational needs grow, at most, quadratic with the number of robots $\times$ the number of measurements per robot, i.e. $\mathcal{O}\left(N^2 m^2\right)$. Technically, the above threshold expresses the case where an operation is needed per different pair of measurements $\{y_a^{(i)}, y_b^{(j)}\}$, with $a, b \in \{1, \ldots, m\}$ and $i, j \in \{1, \ldots, N\}$. Overall, $J(\cdot)$ is evaluated $N+1$ times, one for each robot and one for the global cost function term (5.8), therefore, *STEPS 1-3* are expected to have $\mathcal{O}\left(N^3 m^2\right)$

The computational requirements for *STEP 4*, which is computed on each robot, are dominated by the requirement of solving the least-squares problem (5.12). According to [110, Section 5.5.6, figure 5.5.1], the best algorithms for least-squares problem using SVD procedure, take time that is proportional to $\mathcal{O}\left(T^2 L + L^3\right)$. In the interest of simplicity, and due to the fact that $T \simeq L$, we can assume that *STEP 4* complexity scales as $\mathcal{O}\left(L^3\right)$. Although, there exist no theoretical results for providing the lower bound $\bar{L}$ for the size of the regressor vector, practical investigations on many different applications (e.g. [55,103,111]) indicate that it is sufficient enough

STEP	Complexity	Practical	Comments
1-3	$\mathcal{O}\left(J\left(\mathbf{y}\right)\right)$	$\mathcal{O}\left(N^3 m^2\right)$	Application dependent
4	$\mathcal{O}\left(L^3\right)$	$\mathcal{O}(n^3)$	Least-squares

Table 5.1. *Complexity analysis*

to choose $L \geqslant \bar{L} = 2 \times n$, to adequately tackle the local approximation of J_i. Therefore, it is expected that the computational requirements will grow with $\mathcal{O}(n^3)$. Although this step is repeated for each robot (N times), the distributed nature of the algorithm guarantees that no extra computational needs will be required.

Overall, it is expected that the complexity of computing N times the cost function $J(\cdot)$ dominates the requirement of solving the least-squares problem for one robot. Table 5.1 summarizes the complexity bounds discussed in this subsection.

Remark 3 We close this section by accumulating the free parameters of the proposed algorithm. The set is composed by the number of perturbations M, the total number of utilized monomials L and the time-window T over which the least-squares estimation is taking place. According to Remark 2, the number of perturbations M should be greater than $2 \times n$. Furthermore, the complexity analysis of *STEP 4* indicates that the estimator (5.10) should have at least $\bar{L} = 2 \times n$ number of monomials. Finally, T is a non-negative integer that expresses the desired "forgetting factor" for the constructed estimator. In the following experimental set-ups, we set the algorithm's parameters within these bounds. However, if one wants, all the parameters could be manually tuned in order to achieve better – application dependent – performance.

5.4 Adaptive coverage control utilizing Voronoi partitioning

The first simulation set-up is the well-investigated optimal robots' placement problem [1, 20, 112]. The objective for the network of robots is to spread out over an environment, while aggregating in areas of high sensory interest. Furthermore, the robots do not know beforehand where the areas of sensory interest are, but they learn this information on-line from sensor measurements. The aforementioned task can be found in applications such as environmental monitoring and clean-up, automatic surveillance of rooms/buildings/towns [113–115], or search and rescue [116, 117].

5.4.1 Problem definition

It is assumed that the operational area is a bounded $Q \subset \mathbb{R}^n$. A point inside this environment is denoted by q and the decision vector x_i for the i-th robot contains its position in Q. Also, let $\{V_1, \ldots, V_N\}$ be the Voronoi

partition of Q, for which the robot positions are the generator points:

$$V_i = \{q \in Q | \, \|q - x_i\| \leqslant \left\|q - x_j\right\|, \forall j \neq i\}$$

Let $\zeta(\cdot)$ to be the unknown sensory function such that $\zeta : Q \to \mathbb{R}_{>0}$ (where $\mathbb{R}_{>0}$ is the set of strictly positive real numbers). In other words, this function $\zeta(\cdot)$ assigns in each location of the available space Q a weight of importance, related to the necessity of being covered.

The global cost function for the problem in hand, admits the following form:

$$\mathcal{J}(\mathbf{x}(k)) = \sum_{i=1}^{N} \int_{V_i} \frac{1}{2} \|q - x_i\|^2 \zeta(q) dq \tag{5.15}$$

Apparently, the above function cannot be calculated in advance due to the dependence of the unknown sensory function ζ. Without loss of generality, we assume that the sensory function is given by:

$$\zeta(q) = \mathcal{K}(q)^\tau \upsilon + \mathcal{O}(1/W), \ \forall q \in Q \tag{5.16}$$

where $\mathcal{K} : Q \to \mathbb{R}^W_{>0}$ denotes a vector of bounded, continuous basis functions (e.g. Gaussians, wavelets, sigmoids, etc.) and $\upsilon \in \mathbb{R}^W$ is the parameter vector. The deviation from the actual value of ζ is in the order of the number of basis functions $\mathcal{O}(1/W)$. Although $\mathcal{K}$ is known a priori, the mixing parameters vector υ is environment-dependent and generally unknown. However, the value of the sensory function can be measured from the robots' sensors (e.g. temperature/chemical sensor) at their current position's configuration $\mathbf{x}(k)$.

$$y(x_i) = \zeta(x_i) \tag{5.17}$$

The value of parameter estimation vector $\hat{\upsilon}$ can be approximated through these measurements, utilizing standard parameter estimation techniques [e.g. least-squares approach (5.12)]. Therefore, after the update on the parameter vector $\hat{\upsilon}$, a new update on the belief regarding the sensory function is also available thought the equation:

$$\hat{\zeta} = \mathcal{K}^\tau \hat{\upsilon} \tag{5.18}$$

Hence, the value of the unknown cost function can be approximated through the following equation:

$$J(\mathbf{y}(k)) = \sum_{i=1}^{N} \int_{V_i} \frac{1}{2} \|q - x_i\|^2 \mathcal{K}^\tau(q) \hat{\upsilon} dq \tag{5.19}$$

5.4.2 Simulation results

The operation area was restricted to the plane $[0,1]^2$, so any value, that is afterwards mentioned, has been casted to this context. For implementation reasons, we assume that the operation area consists of 225 discrete points, uniformly distributed across the plane of $[0,1]^2$. The sensory function, $\zeta(q)$, was parameterized as a linear combination of 49 Gaussians, i.e. $\mathcal{K}(j) = \frac{1}{2\pi\sigma_j^2}\exp - \frac{(q-\mu_j)^2}{2\sigma_j^2}$, $\forall j \in \{1,\dots,49\}$. Each standard deviation is set to be $\sigma_j = 0.02$ and the Gaussians centers μ_j are chosen so as to be uniformly distributed in the operational area (seven Gaussians in each row and column). The unknown parameter vector was chosen as $\upsilon = [100, 0.1, 0.1, \dots, 0.1, 100]^\tau$, i.e. the lower left and the upper right Gaussians dominate the sensory function $\zeta(q)$. Finally, the equations are integrated using a fixed step of $\alpha = dt = 0.01$ and the initial values for the estimation of parameter vector (robots' knowledge) was chosen to be $\hat{\upsilon} = [0.1, 0.1, \dots, 0.1]^\tau$.

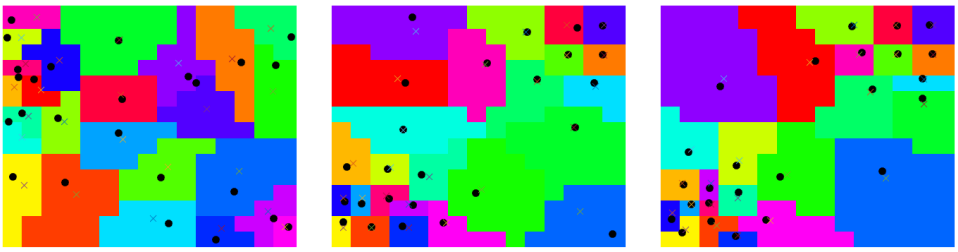
Additionally with the proposed approach, we present simulation results from the algorithm as proposed, for the problem in hand, in [1]. The weights selection was undertaken following the authors' instructions in [1, Section 7.2]. To construct comparable simulations instances, we utilize the same learning rule for the parameter vector $\hat{\upsilon}$ [1, eq. 13]. In both the evaluated algorithms, the update of parameter vector was performed, by aggregating all the robots' measurements. To evaluate the performance of each approach in each timestamp, we also calculate the real value of the cost function (5.15), but none of the evaluated algorithms utilizes this information.

The proposed approach was employed with a constant time-window for the least-squares estimation of $T = 30$ and the number random perturbations was set to be $M = 100$. To approximate each robot's cost function evolution, we utilize a 3rd order monomial estimator with $L = 10$ and using (5.11) we calculate the number of monomials per order to be $L_1 = 2, L_2 = 3$ and $L_3 = 4$.

Random initial positions scenario

In the first simulation scenario, the robots were placed randomly along the x and y axes of the operation area. An example of this simulation set-up is illustrated in figure 5.1, where the figure 5.1(a) illustrates the robots' initial positions and figure 5.1(c) presents the robots final positions, as calculated by the proposed methodology. We mark with 'x' the weighted centroid of each robot's Voronoi partition. As one can see in figure 5.1(c), the robots

gathered around the areas with the highest values of the unknown sensory function $\zeta(\cdot)$.



(a) Initial positions (b) Intermediate stage (c) Final configuration

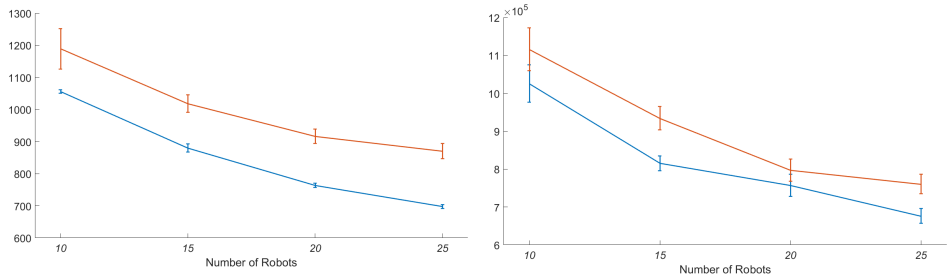
Figure 5.1. *Illustrative example with random initial positions for the robots. In these 3 snapshots is sketched how the proposed algorithm drives the available robots so as to completely cover the space and to aggregate around areas with high sensory interest. The 'x' mark indicates the corresponding weighted centroid of each robot's Voronoi partition.*

Figure 5.2 presents a comparison study between the evaluated algorithms, over different sizes of robot teams. The number of robots was chosen to be 10,15,20 to 25 robots and for each configuration, we performed 20 experiments with randomly selected initial robots' placement. The average, final achieved cost function (5.15) values, along with the corresponding confidence intervals are illustrated to figure 5.2(a). Additionally, we present the summation of the cost function over the course of each simulation pair [figure 5.2(b)]. It must be emphasized that, although the summation of the cost function may be strongly dependent on the initial robots' positions the final achieved value has a small variance around the average value. This feature highlights the ability of the proposed approach to converge to an optimal configuration, independently of the initial conditions.

Right half-plane scenario

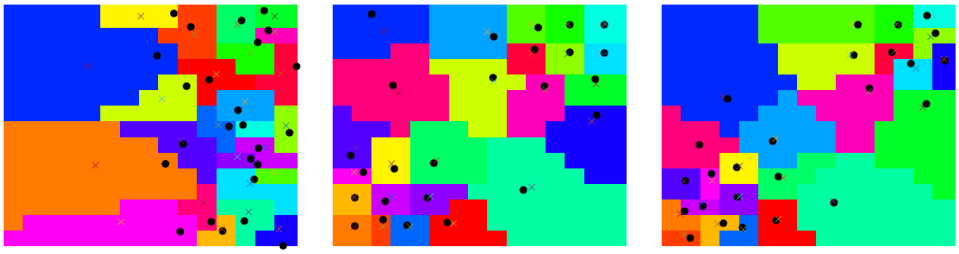
In the second simulation scenario, the robots' initial positions were constrained inside the right half-plane of the operation area. Generally, this scenario has a greater level of difficulty, compared to random initialization, as the robots can easily get stuck in highly sub-optimal situations. Figure 5.3 illustrates an instance of such a scenario where the proposed approach was utilized.

As in the previous scenario, we present a comparison between the evaluated algorithms for different sizes of robot teams. The results are illustrated in figure 5.4. Again, the proposed approach utilizes all the



(a) Final achieved value of the cost function (b) Summation of the cost function over the experiment's horizon

Figure 5.2. *Comparison study for the random initial positions scenario: proposed algorithm (blue) and approach presented in [1] (red).*



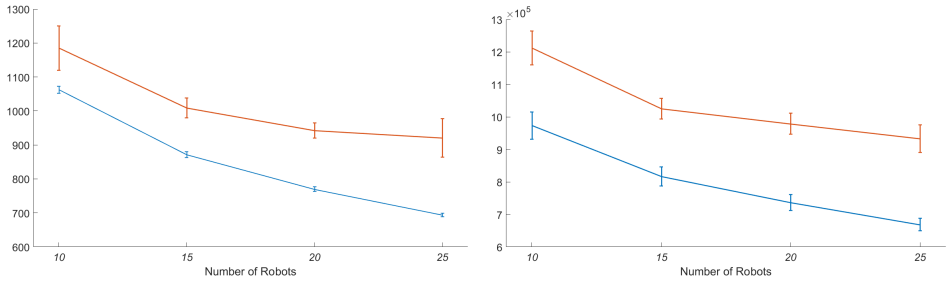
(a) Intial positions (b) Intermediate stage (c) Final configuration

Figure 5.3. *Illustrative example where the robots initial positions are constrained inside the right half-plane of the operational environment. The proposed algorithm navigates the robots around the space, utilizing only their measurements on their current positions, to achieve the mission objective. The 'x' mark indicates the corresponding weighted centroid of each robot's Voronoi partition.*

available team resources in order to achieve optimal robot configurations with small variance around the average values.

5.5 Three dimensional surveillance of unknown areas

A more elaborate variation of the previously described set-up has been proposed in [2] and applied in several domains (e.g [55, 57]). Although the problem is again the optimal placement of robots in real-time, the details of the simulation set-up are important. The terrain to be covered is considered an unknown, non-convex, 3D surface, the formation of which may form an arbitrary number and shape of obstacles. Additionally, we



(a) Final achieved value of the cost function (b) Summation of the cost function over the experiment's horizon

Figure 5.4. *Comparison study for the right half-plane scenario: proposed algorithm (blue) and approach presented in [1] (red).*

employ a realistic model for the robots' sensors (section 4.4.2) and it is utilized in all the simulation scenarios.

5.5.1 Problem definition

The decision variables (5.1) represent the positions of the robots in 3D space, i.e., $\mathbf{x} = \left[x_1^\tau, \ldots, x_N^\tau\right]^\tau$, where $x_i \in \mathbb{R}^3$.

Let us assume that the area to be monitored is constrained within a rectangle in the (x, y)—coordinates as

$$\mathcal{U} = \left\{x, y \mid x \in [x_{min}, x_{max}], y \in [y_{min}, y_{max}]\right\}$$

where $x_{min}, x_{max}, y_{min}, y_{max}$ are real numbers that define the "borders" of the area of interest. Using the definition of $\mathcal{U}$, the area can be defined as a function that maps each point $(x, y) \in \mathcal{U}$ to a point $z = z(x, y)$ [height of unknown terrain at (x, y)]. A point $q = (x, y, z)$ of the terrain is *visible* if there exist at least one robot so that:

- the robot x_i and the point q are connected by a line-of-sight;
- $\|x_i - q\| \leqslant thres$, where *thres* defines the maximum distance the i-th robot can "see".

Given the robots configuration $\mathbf{x}(k)$ at timestamp k, we let $\mathcal{V}$ to denote the *visible* area of the terrain, i.e. $\mathcal{V}$ consists of all points $q \in \mathcal{U}$ that are *visible* from the robots.

Furthermore, the measurements' model for all the robots admits the following form:

$$y_{x_i-q} = \begin{cases} \|x_i - q\| + h_\xi(x_i, q)\xi & \text{if } q \in \mathcal{V} \\ \text{undefined} & \text{otherwise} \end{cases} \quad \forall q \tag{5.20}$$

where $h_\xi(x_i, q)$ is the multiplicative sensor noise term (*e.g.* $\propto \|x_i - q\|^2$) and ξ is a standard Gaussian noise. The above non-linear noise model is a realistic representation of the noise effect in many real robot systems [118] [119, Chapter 3-4]. For instance, in the case of sonar or cameras, the noise affecting such sensors is proportional to the sensor-to-sensing point distance, i.e., the larger is the robot-to-sensing point distance, the larger is the sensor noise [57].

Having the above formulation in mind, we define the following combined cost function that the robot team has to minimize:

$$J(\mathbf{y}(k)) = \int_{q\in\mathcal{V}} \min_{i=1,\ldots,N} y_{x_i-q}\, dq + K \int_{q\in\mathcal{U}\setminus\mathcal{V}} dq \tag{5.21}$$

The first term, in the above equation, is equivalent with the cost function considered in many coverage problems for known 2D environments [20]. The second term is related to the invisible area in the terrain. The positive constant K serves as a weight for giving less or more priority to one of the objectives.

Moreover, the set of non-linear constraints (5.6), which must be hold for each new robots' configuration $\mathbf{x}(k)$, include the following:

- the robots remain within the terrain's limits, i.e. within $[x_{min}, x_{max}]$ and $[y_{min}, y_{max}]$ in the x- and y-axes, respectively;

- the robots satisfy a maximum height requirement, while they do not hit the terrain, i.e. they remain within $[z + d_h, z_{max}]$ along the z-axis, where d_h denotes the minimum safety distance the robots should always have from the terrain and z_{max} denotes the maximum allowable operational height for the robots;

- $\left\|x_i - x_j\right\| \geqslant d_r,\ \forall i, j \in \{1, \ldots, N\}$ and $i \neq j$, i.e. the safety distance between two robots is d_r.

5.5.2 Simulation results

For comparison purposes, we utilize the centralized CAO-based approach that has been proposed for the problem in hand [2]. The proposed approach was parametrized with a time-window $T = 40$ for the least-squares estimation, with $M = 100$ random perturbations, the corresponding approximator was a 3rd order monomial estimator with $L = 18$, and the number of monomials per order (5.11) were $L_1 = 2, L_2 = 5$ and $L_3 = 10$. Acknowledging the fact that, the CAO algorithm performs optimization in a higher dimension space, a different set of parameters was chosen. Evaluating the CAO version for different number of random perturbation, we found that after $M = 900$ the number of random perturbation does not affect its performance. Furthermore, to cope with the higher dimension state-space, the time-window was set to $T = 60$ and the approximator was chosen to be 3rd order monomial estimator with $L_1 = 3, L_2 = 12$ and $L_3 = 40$ (with overall size of $L = 56$). In both the algorithms, we utilize $\alpha = 0.1$ to update the robot's positions. For the rest of this section, we use these values in all the presented experiments.

To perform simulations in a realistic environment, we utilized an area located in Zürich, Switzerland [figure 5.5(a)]. This map was generated using a state-of-the-art visual-SLAM algorithm [120], which tracks the pose of the camera while, simultaneously and autonomously, building an incremental map of the surrounding environment. The dimension of the terrain was squeezed to be inside of $[0, 80]$ for x- and y-axes, while the height of the terrain was between $[0, 7.2]$m and the maximum operational height was set to 25m. Following the authors instructions [2], K weight (5.21) was chosen to be 30, while both the safety distance from the terrain and the minimum allowable distance between two robots were set to be $d_h = d_r = 0.5$m. Finally, the duration of each experiment was set to $k_{max} = 600$ timestamps.

Figure 5.5 depicts such a simulation instance with 6 robots. The initial positions of the robots, as it is sketched in figure 5.5(b), were selected to be "crowded" inside a sub-area of the terrain. Figures 5.5(c) and 5.5(d) illustrate the final robots' configuration, as calculated by the CAO-based algorithm in 3D and 2D representation, respectively. The corresponding final robots' assignment as calculated by the proposed approach is presented in figures 5.5(e) and 5.5(f). In both cases, the 3D representation reports which sub-area of the terrain is cover by each robot, while the 2D representation reveals the exact positions of the robots in x-y plane and the distance between them. Figure 5.5(g) depicts the evolution

of the cost function (5.21) for both the evaluated algorithms. Apart from the difference in the convergent state, the proposed approach is able to find this solution from its early steps (< 50). The centralized CAO needs more iterations to learn the dynamics of the robots and the unknown terrain, because it performs its optimization scheme in the higher dimension space of $\mathbb{R}^{3N}$. On the contrary, the proposed algorithm separately – although cooperatively – solves N optimization problems of the size of $\mathbb{R}^3$.

In the specific problem set-up, the speed of convergence requires extra attention, as a slow convergence rate may lead to instability or loss of convergence at all. More specifically, if a navigation algorithm does not converge fast enough to the optimal configuration, one or more robots may have reached high-altitude positions, from which they cannot acquire useful measurements [out of their sensor capabilities (5.20)]. This is a non-recoverable situation, as the robots do not have any "feedback" from the terrain to properly evaluate their actions.

Scalability analysis

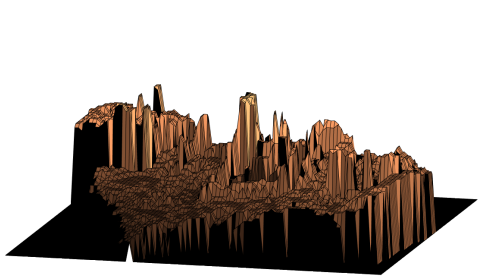
To validate both the efficiency and the effectiveness of the proposed algorithm in case of bigger robot teams, we performed experiments with 5,10,15 and 20 robots[9]. For each different size of robotic team, we create 20 experiments instances with randomly chosen initial robots' positions. The aforementioned simulations instances are evaluated on both the proposed approach and the centralized CAO-based one.

The results of these simulations are summarized in figure 5.6. Figure 5.6(a) displays the average value of the resulting cost function $J(k_{max})$, along with the corresponding confidence interval, over the different number of robots. Additionally, figure 5.6(b) displays a statistical analysis on the summation of cost function $\sum_0^{k_{max}} J(k)$, to investigate the convergence rate of each pair (scenario-algorithm).

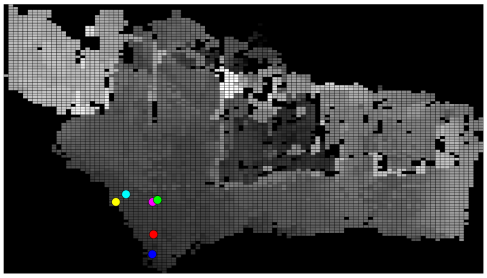
Overall, the proposed approach achieves an average improvement of 23% on the *final achieved cost function value*, with 55.33% improvement on the deviation around that average value. Moreover, the *summation of cost function* has been improved by 23.84% with a corresponding improvement on the deviation of 65.06%, against the centralized CAO-based approach. The proposed approach achieves these performance enhancements mainly due to the two following reasons:

(i) the proposed algorithm has a better perspective on the change of

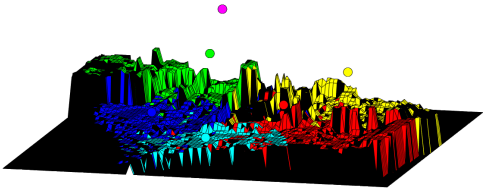
[9]Please note that, for the current experiment set-up with the previously defined sensor's capabilities, the utilization of more than 15 robots cannot significantly affect the coverage task.



(a) 3D representation of the surface to be covered

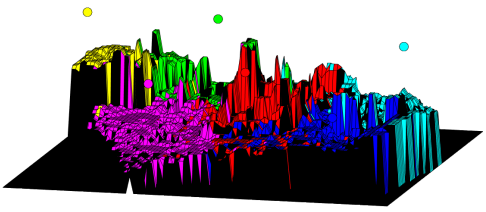


(b) Initial positions of the available robots

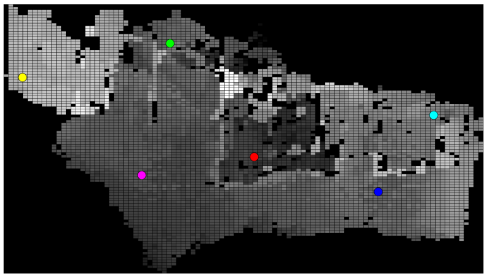


(c) 3D view - CAO-based approach

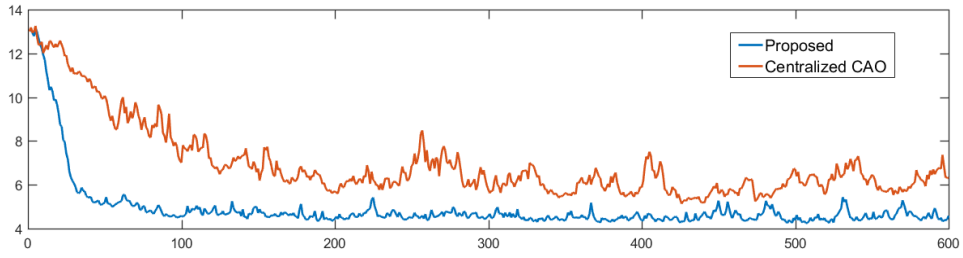
(d) Top view - CAO-based approach



(e) 3D view - proposed approach

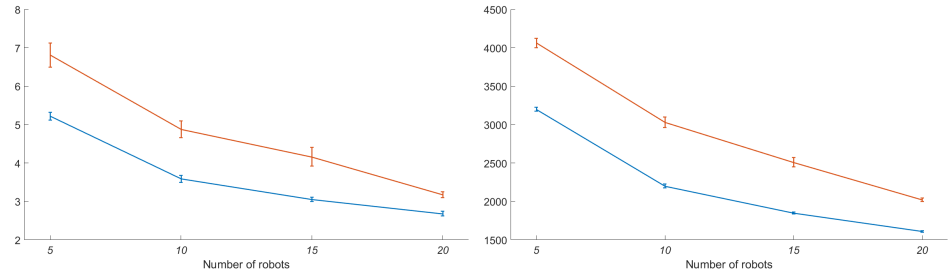


(f) Top view - proposed approach



(g) Cost function evolution

Figure 5.5. *Indicative example: surveillance of unknown terrain by a team of robots. The proposed algorithm and the CAO-based approach [2] are evaluated on the same set-up (environment, robots initial positions, robots sensor capabilities).*



(a) Final achieved value of the cost function (b) Summation of the cost function over the experiment's horizon

Figure 5.6. *Comparison study over different number of robots: proposed algorithm (blue) and CAO-based approach [2] (red).*

the overall cost function by evaluating the appropriate combinations of historical measurements on that cost function [(*STEP 2-3*) of the proposed approach].

(ii) the fast convergence of the proposed approach eliminates the chances for a robot to be found out of its sensors capabilities. Therefore, the proposed approach is able to converge on approximately the same robots' configuration (per different team size), independently on the robots' initial positions. The latter is depicted on the substantial improvements on the corresponding confidence intervals.

Fault tolerant characteristics

In this scenario, we investigate the performance of the proposed algorithm in case of catastrophic events or hardware failures. More precisely, 5 robots were initially deployed to perform the aforementioned coverage task, while the duration of the experiment was increased to k_{max} =1000 timestamps. It is assumed that, at timestamp 330, one robot didn't correspond to our control commands and the measurements' flow had been interrupted. Under these new circumstances, the surveillance task has to be undertaken by remaining, properly working robots. After the completion of the 2/3 of the available timestamps, we assume that another robot had an equipment malfunction and cannot continue its covering task. Thus, the number of available robots, which are called to cover the area of interest for the ~300 remaining timestamps, has dropped to 3.

Figures 5.7(a)-5.7(f) illustrate the evolution of the robots' positions during the course of the previously described scenario, utilizing the proposed approach. After both the robots' malfunctions, the algorithm redesigns

the remaining robot positions to achieve the best possible coverage. Overall, figure 5.7(g) demonstrates the evolution of the objective function for the proposed approach in comparison with the centralized CAO-based approach [2].

It must be emphasized, that the proposed algorithm does not need any separately designed, fault detection mechanism (e.g. failure in establishing communication, user detected, etc.), as it is able to implicitly derive this kind of information from the changes in the cost function J with respect to the commanded positions. The above feature is of paramount importance in real-life multi-robots' applications, since it removes the tedious – and in many applications impossible – task to predict all the possible malfunctions, as well as to design the appropriate course of actions.

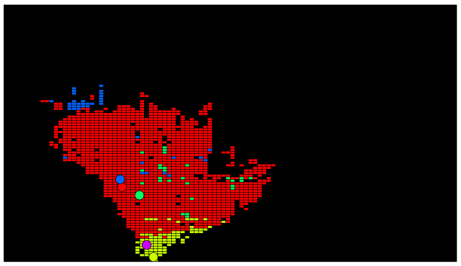
Target monitoring

We close this section by investigating the algorithm's capability to process objectives that can be alternated/activated on-the-fly, without stopping and restarting the mission. To achieve that, simultaneously with the coverage task, we introduce the task of monitoring a target. For the sake of this simulation set-up, it is assumed that – in addition to the sensors which are responsible for the coverage task (5.20) – the robots are equipped with exteroceptive sensors (e.g. cameras, sonars, etc) which are able to estimate the targets' positions, according to the following measurement model:

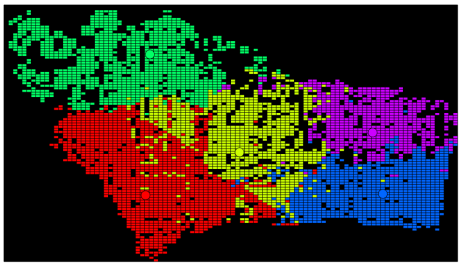
$$y_{x_i-\chi_j} = \begin{cases} \left\|x_i-\chi_j\right\| + h_\xi(x_i,\chi_j)\xi & \text{if } \chi_j \textit{ has} \\ & \textit{been detected} \\ \text{undefined} & \text{otherwise} \end{cases} \tag{5.22}$$

where χ_j denotes the j-th target's position in 3D space, $h_\xi(x_i, x_t)$ and ξ – similar to equation (5.20) – denote the multiplicative sensor noise term and the standard Gaussian noise, respectively. Therefore, an extra term has to be added to the cost function (5.21) to appropriately evaluate the progress of targets' monitoring, as follows:

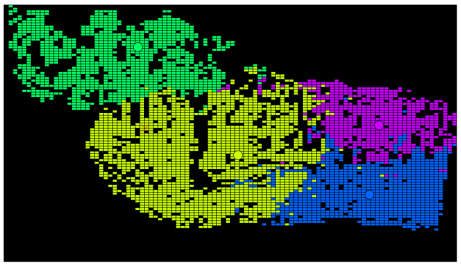
$$\begin{aligned} J(\mathbf{y}(k)) = \int_{q\in\mathcal{V}} \min_{i=1,\dots,N} y_{x_i-q}\,dq + K\int_{q\in\mathcal{U}\setminus\mathcal{V}} dq \\ + K_t \sum_{j=1}^{n_t} \min_{i=1,\dots,N} y_{x_i-\chi_j} \end{aligned} \tag{5.23}$$



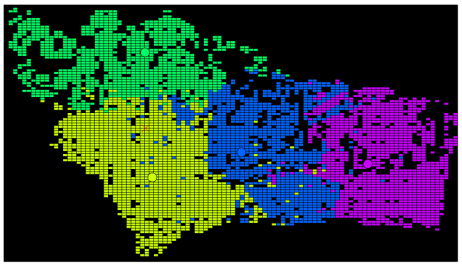
(a) Initial positions of the 5 available robots



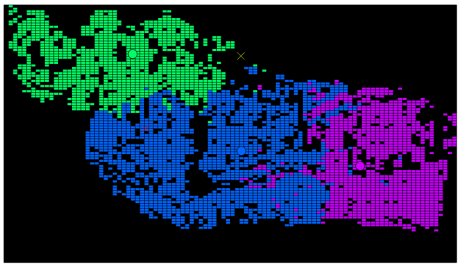
(b) Coverage task with all 5 available robots



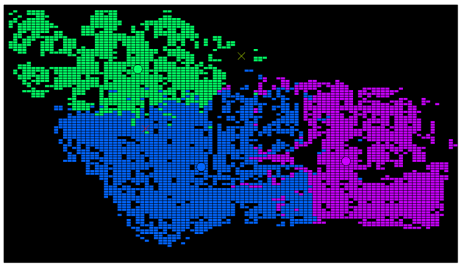
(c) One timestamp after the malfunction on the red robot



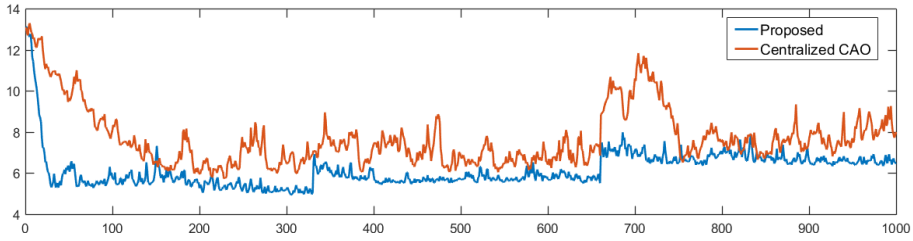
(d) Coverage task with 4 robots



(e) One timestamp after the malfunction on the yellow robot



(f) Again, the algorithm redisgns the robots positions to cover the area in the best possible way utilizing the available resources



(g) Cost function evolution

Figure 5.7. *Malfunction scenario: 5 robots were initially deployed for the surveillance task. At two distinct timestamps, the swarm of robots loses one of its member due to a simulated malfunction. The surveillance task have to be continued with the remaining team resources.*

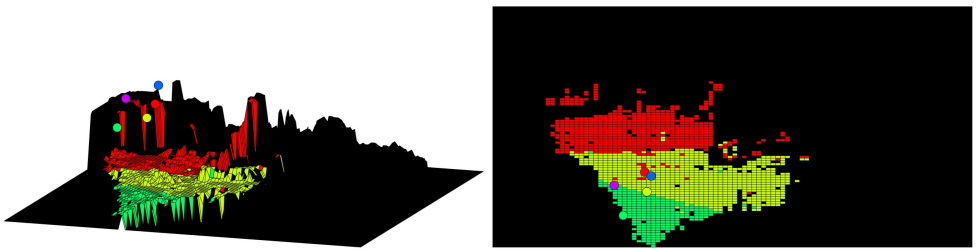
where K_t serves as weight to give more or less priority to the monitoring task in comparison with the coverage. In addition, n_t denotes the number of targets to be monitored.

The experiments were performed in the same terrain, under the previously defined set-up parameters. Figure 5.8 illustrates four key-snapshots, which demonstrate the functionality of the proposed algorithm. Figure 5.8(a) depicts the robots' initial positions along with the corresponding coverage on the terrain. After 367 timestamps [figure 5.8(b)], the algorithm has converged to the (locally) optimal robots' configuration for the coverage – only – problem. At $k = 370$ timestamp, it is assumed that a target, which requires closer examination, appears inside the operation area. The proposed algorithm, after the time needed to learn the changed problem dynamics [activation of the third term in (5.23)], starts to adapt the robots' positions to minimize the updated cost function (5.23). More precisely, as illustrated in figure 5.8(c), the purple robot (which was, at the time, closer to the target) starts to gain height to minimize its distance from the detected target. However, such an action leads to poor coverage on the sub-area underneath that robot. To tackle the above undesirable situation, the proposed algorithm redesigns the remaining robots' positions so as to achieve the best coverage of the terrain with the available resources. The final robots' positions with the corresponding coverage of the terrain is sketched in figure 5.8(d). The evolution of the objective function for the proposed approach in comparison with the centralized CAO-based approach is demonstrated in figure 5.9. Conclusively, for this simulation scenario, the proposed algorithm

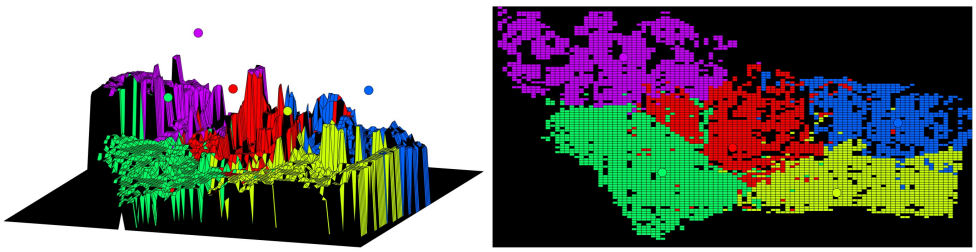
- chooses to assign a robot to be as close as possible to the target without any explicit command,

- adapts the other robots' positions so as to "fill the hole" in the coverage task, and

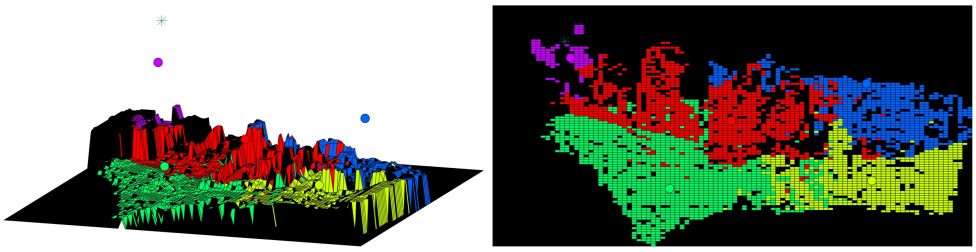
- achieves almost the same level of terrain coverage with the centralized CAO-based approach for 5 robots [figure (5.9) dashed line], while 1 (out of 5) robots is occupied with other task.



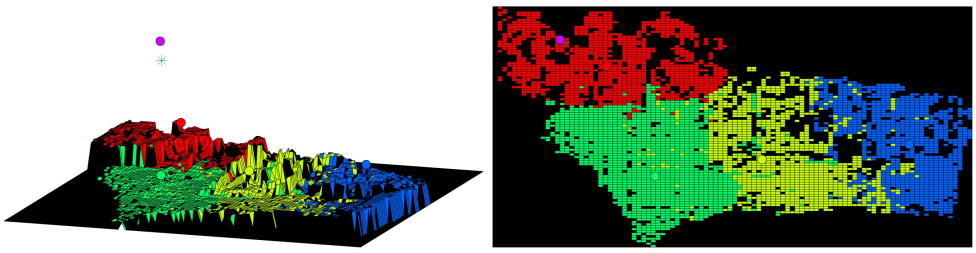
(a) Timestamp 1 - Initial positions of the 5 available robots



(b) Timestamp 367 - Coverage task with all 5 available robots



(c) Timestamp 427 - The purple robot starts to gain height to minimize the distance from the target. As a consequence, it cannot cover adequately its underneath surface



(d) Timestamp 1000 - Finally, the algorithm redesigns the robots positions so as to cover the area in the best possible way utilizing the available resources

Figure 5.8. *Target monitoring scenario: The robots have been deployed having as extra objective (apart from the surveillance task) to get as close as possible to a target. The target appears inside the operation area of the robots in the middle of the mission.*

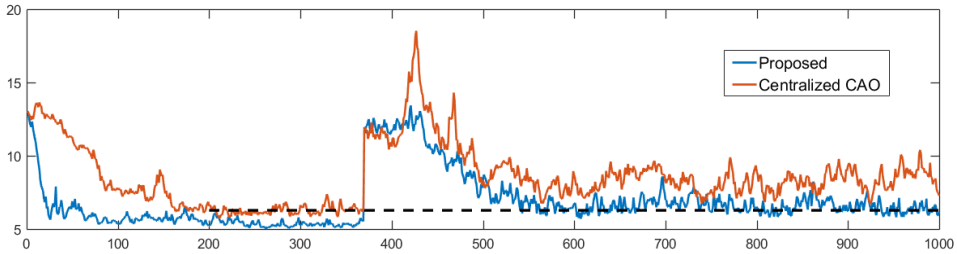


Figure 5.9. *Cost function evolution in target monitoring scenario.*

5.6 Time-varying formation control

5.6.1 Problem definition

In this application we focus on the problem of formation control in the context of multi-robot systems. More specifically, the objective in this problem set-up is to drive the robots so as to follow a desired trajectory, while maintaining a commanded formation. In the this sub-section, we state the basic definitions for the *time-varying formation control* problem that will be useful in this application. More thorough discussions and definitions around this problem can be found in [105].

The decision variables (5.1) represent the collective vector of all the robots' positions, i.e. $\mathbf{x} = \left[x_1^\tau, \ldots, x_N^\tau\right]^\tau$, where $x_i \in \mathbb{R}^n$.

Let $\sigma_1(\mathbf{x})$ to denote the *centroid* of the swarm of robots, i.e.

$$\sigma_1(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} x_i = B_1\mathbf{x} \tag{5.24}$$

where $B_1 \in \mathbb{R}^{n\times Nn}$ is defined as:

$$B_1 = \frac{1}{N}\left(1_N^\tau \otimes I_n\right)$$

where I_n is the $(n \times n)$ identity matrix, 1_N denotes the all-one vector with N elements and the symbol $\otimes$ represents the Kronecker product.

Also, the *formation* of the system is denoted as a set of relative displacements between the robots:

$$\sigma_2(\mathbf{x}) = [(x_2 - x_1)^\tau (x_3 - x_2)^\tau \ldots (x_N - x_{N-1})^\tau]^\tau = B_2\mathbf{x} \tag{5.25}$$

where $B_2 \in \mathbb{R}^{(N-1)Xffl \times NXffl}$ is defined as:

$$B_2 = \begin{bmatrix} -I_n & I_n & O_n & \dots & O_n & O_n \\ O_n & -I_n & I_n & \dots & O_n & O_n \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ O_n & O_n & O_n & \dots & -I_n & I_n \end{bmatrix}$$

where O_n denotes the $(n \times n)$ null matrix. Additionally, let $\sigma_{1,d}$ and $\sigma_{2,d}$ to denote the desirable (user defined) centroid and formation of the swarm of the robots, respectively. In other words, the goal of the problem is translated to design the robots positions, such that the system's σ_1 and σ_2 converges, respectively to $\sigma_{1,d}$ and $\sigma_{2,d}$. The particular form for the cost function that realizes such a reasoning is as follows:

$$J(\mathbf{y}(k)) = \|\sigma_{1,d}(k) - \sigma_1(\mathbf{y}(k))\| + K\|\sigma_{2,d}(k) - \sigma_2(\mathbf{y}(k))\| \tag{5.26}$$

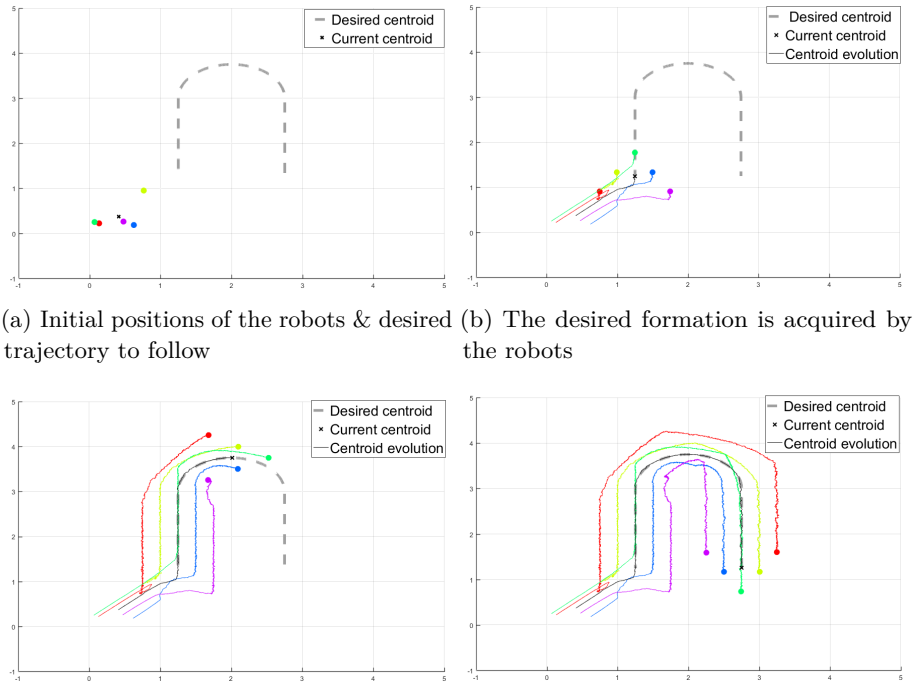
where $\mathbf{y}(k) = \bar{\mathbf{x}}(k) \in \mathbb{R}^n$ and denotes the estimation of robot's position (e.g. GPS measurements) for the k-th timestamp and K is a user defined positive constant.

5.6.2 Simulation results

For comparison purposes, we adopt the simulation environment as described in [105, Section 7.1]. The details of this simulation scenario are described below:

- The robots positions along with the desired paths are defined inside the 2D space, i.e. $n = 2$.
- The multi-robot system is composed of $N = 5$ robots.
- The scenario for this experiment requires for the centroid of the robots (5.24) to move along a desired U-shape path, while the robots retain a V-shape formation.
- The initial and final positions of the U-shaped trajectories (centroid) are $[1.25,\ 1.25]^\tau$m and $[1.25,\ 2.75]^\tau$m, respectively.
- The overall length of the path covered by the robot's centroid is about 8.2m

The proposed approach was employed with a constant time-window for the LS estimation of $T = 8$ and the number random perturbations was set to be $M = 100$. To solve the underlying least-squares optimization problem (5.12), we utilize only a 2rd order monomial estimator with $L_1 = 2$ and $L_2 = 2$ (with overall size of $L = 5$).



(a) Initial positions of the robots & desired trajectory to follow

(b) The desired formation is acquired by the robots

(c) The robots adjust their positions to maintain the desired formation during the course of the experiment

(d) Final positions of the robots along with the evolution of their trajectories

Figure 5.10. *Time-varying formation control for a swarm of 5 robots*

Figure 5.10 illustrates the performance of the proposed algorithm in this simulation scenario. Sub-figure 5.10(a) presents the initial positions of the 5 robots (colored circles), the corresponding centroid (marked with 'x') and the desired trajectory of the centroid (grey dashed line). Sub-figure 5.10(b) focuses on the actions that each robot made in order to take place for the upcoming movement. Sub-figure 5.10(c) and 5.10(c) present an intermediate and the final positions of the robots, respectively. The robots successfully completed the assigned mission by keeping the trajectory of the centroid (black line) close to the desired one and, at the same time, retaining the desired formation.

Although the proposed algorithm cannot reach the performance of a

dedicated approach [105, Fig. 3], due to the fact it learns the time-varying cost function (5.26) from scratch, it provides several advantages. At first, it can straightforwardly incorporate operational constraints (e.g. limited communications, obstacle avoidance functionality, etc.). Secondly and most importantly, the proposed approach does not utilize the desired (user-defined) *centroid* ($\sigma_{1,\mathrm{d}}$) or *formation* ($\sigma_{2,\mathrm{d}}$) inside its decision making process, but only a cost function measurement per different configuration. Overall, the universality of the proposed algorithm allows the realization of any other evaluation scheme (5.24)-(5.26), without the need to explicitly redesign the whole navigation scheme.

5.7 Persistent coverage inside unknown environment

In the final application, we focus on the problem of persistent coverage an area of interest with a swarm of robots. In this application, it is assumed that the operational robots are equipped with the appropriate sensors that are able to cover a portion of the environment. The aim of persistent coverage is to maintain a desired level of coverage over an environment with a time-decaying coverage. The problem along with a specifically designed algorithm has been proposed in [49]. The authors also established a well-defined, heuristic mechanism to on-line share the coverage evolution between the robots in distributed way.

Although the results are remarkable, the proposed decision making mechanism in [49] utilizes a model that accurately predicts the improvement in the coverage level due to the robots movement [49, equations (10),(18)-(21)]. In real-world applications, the above assumption does not always hold, as the increase in coverage level (i) is usually corrupted by non-linear noise, (ii) can be affected by environmental specific characteristics, such as local morphology, obstacles, other robots' positions, etc., (iii) may follow a time-varying model (e.g. coverage level deteriorates over time). To circumvent these difficulties, we propose a variation of the above problem, where the coverage increase, due to robots' movement, cannot be calculated before the action. The actual information about the exact covered area is only available after the execution of each corresponding action through the robot's measurements. The above formulation is not only more *realistic*, as it does not require an exact model of the environment or robot's coverage capabilities, but also more *generic* as it does not need to redesign the approach when robots with different or unknown coverage models are deployed.

5.7.1 Problem definition

It is assumed that the operational area is a bounded $Q \subset \mathbb{R}^2$, which a team of robots has to persistently cover. The decision variables (5.1) represent the collective vector of all the robots' positions, i.e. $\mathbf{x} = \left[x_1^\tau, \ldots, x_N^\tau\right]^\tau$, where $x_i \in Q$.

Inside the environment there are several positions $q \in O \subset Q$ that cannot be traversed by the robots and additionally the presence of these obstacles affects each robot's coverage distribution. Although the exact positions of the obstacles are generally unknown, we assume that the robots are able to sense their presence when they are at close proximity. The above assumption is in line with the most commercial robots which are also equipped with proximity sensors to avoid collisions [121–123]. Thus, each robot's new candidate position x_i^{cand} should verify the following constraint [see equation (5.6) of general Problem Formulation]:

$$\min_{q \in O} \left(\left\| x_i^{\text{cand}} - q \right\| \right) \geqslant b \tag{5.27}$$

where b denotes the safety distance. At each timestamp k, the overall coverage increase is given by $y(q,k) = \sum_{i \in \{1,\ldots,N\}} y_i(q,k)$, $\forall q \in Q$, where

$$y_i(q,k) = \begin{cases} \gamma_i(q, x_i) & \text{if } \|x_i - q\| \leqslant r_i^{\text{cov}} \ \& \\ & \text{there is line-of-sight} \\ & \text{between } x_i \text{ and } q \\ 0 & \text{otherwise} \end{cases} \tag{5.28}$$

and $\gamma_i(q, x_i)$ denotes a non-linear function that models how the coverage level evolves in the area around the i-th robot's position. Please note, that coverage distribution model $\gamma_i(q, x_i)$ may be different and arbitrary defined for each robot, as it depicts the functionality of its on-board sensors.

The coverage of the operational area can be modeled by a time-varying field and, in general, admits the following form:

$$Z(q,k) = d(q)Z(q,k-1) + y(q,k), \ \forall q \in Q \tag{5.29}$$

In other words, the coverage level decreases to a constant decay gain $d(q)$, with $0 < d(q) < 1$, and increases according to the $y(q,k)$. The objective of the multi-robot team is to maintain a desired coverage level, $Z^*(q) > 0$, $\forall q \in Q$.

Having the above formulation in mind, we define the *quadratic coverage error* the robot team has to minimize:

$$J(k) = \int_Q \left(Z^*(q) - Z(q,k)\right)^2 dq \tag{5.30}$$

5.7.2 Simulation results

All simulations were performed in a rectangle environment consisting of 100×150 units, with uniformly distributed decay rate $d(q) = 0.995,\ \forall q \in Q$. The desired coverage level is $Z^*(q) = 100,\ \forall q \in Q$. The number of robots was $N = 6$, while their maximum motion is $u^{max} = 5$. The increase in open-space coverage, caused by the robots' movements, can be simulated by:

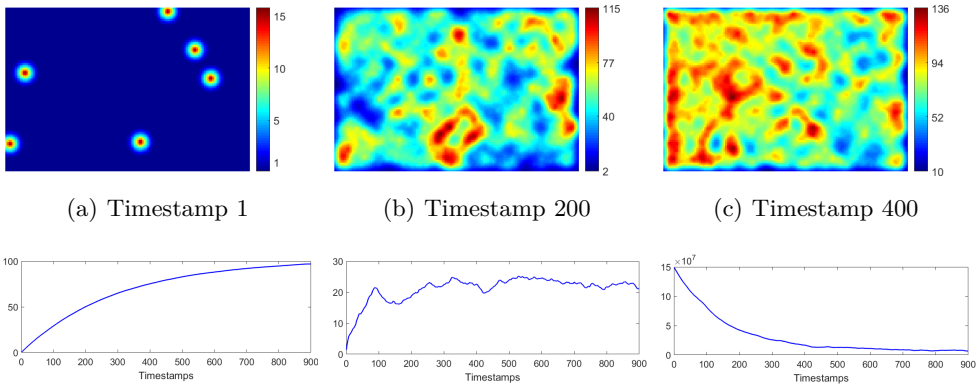
$$\gamma_i(q,k) = \frac{P}{\left(r_i^{cov}\right)^2}\left(\|x_i - q\| - r_i^{cov}\right)^2 \tag{5.31}$$

The maximum value is set to $P = 17$ and the coverage radius is set to $r_i^{cov} = 10$ units. Please note that, this equation is not utilized during the decision-making process, but it is only employed to simulate the increase in the area coverage, due to the robot's movement. Finally the experiments' duration is set to $k_{max} = 900$ timestamps.

To adapt the parameters of the proposed algorithm to the current application, we have to take into consideration that the navigation algorithm has to rapidly change its behavior due to the time-varying nature of the cost function. Therefore, the time-window for the least-squares estimation was only $T = 5$ timestamps and the number of perturbation was $M = 100$ candidates. To solve the underlying least-squares optimization problem (5.12) with such a reduced historical values, we utilize only a 2rd order monomial estimator with $L_1 = 2$ and $L_2 = 2$ (with overall size of $L = 5$). Finally, following also the problem definition in [49, Section II.], we utilize $\alpha = 1$ to update the robot's positions.

Obstacle-free environment

In the first simulation scenario, we deploy the swarm of robots in an obstacle free environment. An indicative simulation run of this scenario is summarized in figure 5.11. Figures 5.11(a)-5.11(c) present the evolution of the coverage across the environment Q, for 3 different timestamps. Additionally, figures 5.11(d), 5.11(e) and 5.11(f) depict the evolution of the *average coverage level*, the corresponding *standard deviation* and the *quadratic coverage error* for the course of the experiment, respectively.



(a) Timestamp 1 (b) Timestamp 200 (c) Timestamp 400

(d) Average coverage level (e) Standard deviation of coverage level (f) Cost function - quadratic coverage error

Figure 5.11. *Obstacle-free scenario: Figures (a)-(c) illustrate the coverage level for 3 different timestamps and figures (d)-(f) depict the corresponding performance indices*

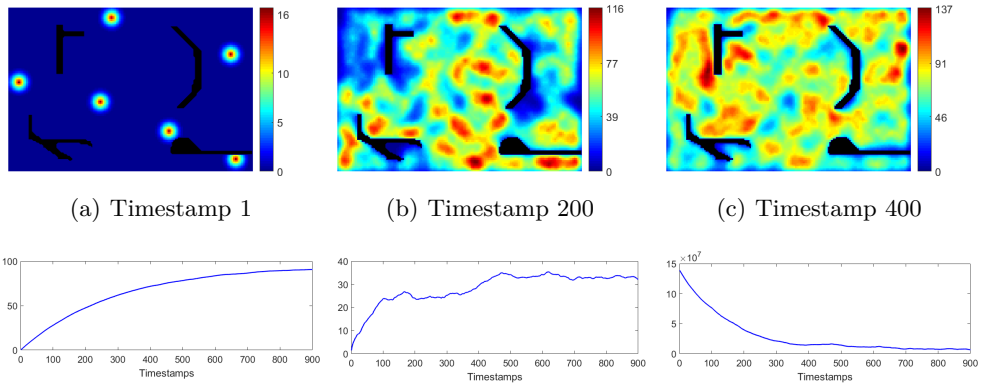
After the experiment execution, the average coverage level in all the operational environment Q was 97 with a standard deviation of 21.2 and the corresponding quadratic coverage error was 6.9×10^6.

It should be highlighted that, the objective (5.30) is a time-varying function with high rate of change, i.e. the evaluation of (5.30) may result in significantly different scores for the same robots positions, even for very close timestamps. However, the proposed scheme is able to appropriately tackle the above problem, by constantly learning these cost function variations with respect to the robots' positions.

Although the proposed algorithm presents an equivalent performance compared to the dedicated one [49, Section VI.], if the problem is defined as in this scenario and the coverage increase due to the robots movement can be accurately predicted, a dedicated approach should be preferred to avoid the extra time due to learning [equations (5.10),(5.12) of the proposed algorithm]. However, the proposed approach has several advantages when it is deployed in a real-world environment, where the evaluation of the coverage increase cannot be performed beforehand. Such a scenario is presented in the following paragraph.

Unknown cluttered environment

In the final simulation scenario, we investigate the performance of the proposed approach for the persistent coverage task, when it is evaluated on an unknown environment with non-convex obstacles. The obstacles



(a) Timestamp 1 (b) Timestamp 200 (c) Timestamp 400

(d) Average coverage level (e) Standard deviation of coverage level (f) Cost function - quadratic coverage error

Figure 5.12. *Scenario in unknown environment with non-convex obstacles: Figures (a)-(c) illustrate the coverage level for 3 different timestamps and figures (d)-(f) depict the corresponding performance indices*

have been created randomly and don't hold any kind of pattern. The minimum distance between the obstacles and any robot (5.27) has been set to $\mathsf{b} = 2.5$.

Again, an illustrative example is presented in figure 5.12. Figures 5.12(a), 5.12(b), 5.12(c) present the evolution of the of the coverage across the environment Q, for 3 different timestamps. Figures 5.12(d), 5.12(e) and 5.12(f) depict the evolution of the *average coverage level*, the corresponding *standard deviation* and the *quadratic coverage error* (5.30), respectively.

The cost function (5.30) does not need any adaptation to this scenario as the coverage values $\mathsf{Z}(\mathsf{q})$ that correspond to obstructed locations $\mathsf{q} \in \mathsf{O}$ will remain zero, independently of their distance from any robot. In other words, the calculation of (5.30) does not need the information of the unknown obstacles, as the robots would never send coverage updates (5.28) about the obstacles' positions. However, to construct comparable metrics with the previous scenario, we exclude the values that correspond to obstacles' locations from the calculation of the average coverage level (figure 5.12(d)). After the experiment execution, the average coverage level inside Q was 90.7 with a standard deviation of 32.1 and the corresponding quadratic coverage error was 6.6×10^6.

Comparing the outcomes of two scenarios side by side, we can draw the following observations:

- In the cluttered environment scenario, the robots can more easily

get "trapped" in overcovered areas, resulting in a higher standard deviation. In other words, when a robot detects (implicitly from the changes in its corresponding cost function) that its position deteriorates the coverage level, may have only a small sub-set of possible new positions.

- During the course of the experiment in the cluttered environment, the obstacles "blocked" a portion of the robots' coverage capabilities. Therefore, for the cluttered environment scenario, the robots achieved a smaller average coverage level (excluding the obstacles positions).

5.8 Conclusions

A distributed methodology for dealing multi-robot problems, where the mission objectives can be translated to an optimization of a cost function, has been proposed. Contrary to the majority of the multi-robot approaches, where the objectives are accomplished in cost function optimization scheme, the proposed approach is designed for multi-robot problems where the a priori calculation of the cost function is not feasible. In a nutshell, the proposed approach has the following key advantages:

- it does not require any knowledge of the dynamics of the overall system;
- it can incorporate any kind of operational constraint or physical limitation;
- it shares the same convergence characteristics as those of block coordinate descent algorithms;
- it has fault tolerant characteristics;
- it can appropriately tackle time-varying cost functions;
- and it can be realized in embedded systems with limited power resources.

Conclusively, we expect that many interesting tasks in mobile robotics can be approached by the proposed scheme. This is basically due to the fact that the proposed approach, instead of explicitly solving a particular problem, which requires prior knowledge of the system dynamics, learns – from the real-time measurements – exactly the features of the system which affects the user-defined objectives. Furthermore, the proposed approach

can be appealing in many real-life applications due to its fault-tolerant characteristics, without an explicitly designed fault-detection mechanism. All the above issues are considered of paramount importance in multi-robot applications.

As future directions, we are interested in performing an extensive set of experiments, ideally with a large number of robots (e.g. a large swarm of femtosatellites (100-gram-class spacecraft) [124]). In particular, in such a set-up, it is impossible to explicitly program each and every robot to perform a sub-task, therefore the goal will be to achieve an abstract set of objectives, which are defined in form of cost function optimization. The idea behind the above formulation is, by excluding the intermediate steps from the design process, we enrich the multi-robot decision making scheme with autonomy, regarding the "type" of converged solutions.

6 Conclusion

Contents

In this Thesis we have addressed the problem of navigating a team of robots so as to achieve several team objectives. This is undoubtedly a quite important task with several real-word applications, an example of which has been demonstrated in section 4.6. Current top performing approaches, either attack a simplified version of the original problem or rely on heavy simulations to tune a parametrized decision-making mechanism.

The contributions of this Thesis can be summarized as follows:

- A new algorithm for the problem of multi-Robot Coverage Path Planning (mCPP) is proposed, which is based on spanning tree coverage (STC) framework. First, the available space of is divided into distinct classes, as many as the number of robots. 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 by taking 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. Second, 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.

- A novel method for dealing the problem of exploring an unknown area using multi-robot teams under environmental and communication constraints, while simultaneously building a detailed map of the environment has been proposed. Based on this approach, we are transforming a standard trajectory generation problem so as to optimize a transformed version of trajectory generation efficiency, employing CAO algorithm. The methodology proposed is independent of requirements regarding operational characteristics of the robots like communication range and type of sensors. Additionally, the proposed scheme, in relation to the vast majority of the optimal/dynamic programming approaches, takes into account the non-linear characteristics of the robots' sensors and the fact that the operation area is unknown. In a nutshell, the proposed methodology aims to bridge the gap between the state-of-the-art algorithms and the actual practices, by successfully navigating the robots through environments, where the objective/reward function cannot be calculated a-priori, due to the aforementioned reasons. The applicability and adaptability of our approach in realistic scenarios has been demonstrated through simulated and real-life underwater sea-floor mapping experiments in the port of Porto, Portugal using a team of AUVs.

- Building on the previously described CAO methodology, we propose a general-purpose, distributed methodology for dealing multi-robot problems, where the mission objectives can be translated to an optimization of a cost function. Contrary to the majority of the multi-robot approaches, where the objectives are accomplished in cost function optimization scheme, the proposed approach is designed for multi-robot problems where the a priori calculation of the cost function is not feasible. In a nutshell, the proposed approach has the following key advantages:

  - it does not require any knowledge of the dynamics of the overall system;
  - it can incorporate any kind of operational constraint or physical limitation;

- it shares the same convergence characteristics as those of block coordinate descent algorithms;
- it has fault tolerant characteristics;
- it can appropriately tackle time-varying cost functions;
- and it can be realized in embedded systems with limited power resources.

The proposed algorithm is evaluated in four heterogeneous simulation set-ups under multiple scenarios, against both general purpose (centralized) and specifically-tailored to the problem in hand, algorithms.

6.1 Future directions

Several avenues of exploration are left open for future work, for the case of offline multi-robot Coverage Path Planning problem (DARP algorithm). 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.

The proposed approach, for the case of online trajectory generation, achieves state-of-the-art performance in the multiple evaluated set-ups. Naturally, further improvements are possible. Usually, in multi-robot frameworks, prior information about the objectives and mission are available before the actual deployment of the robots. The exploitation of this information can provide an initial plan for the operational robots. The advantages of such prior step could be twofold. First, the robots will be ready to accomplish the desirable objectives from the early stages of the mission. Second, incorporating the offline knowledge about the mission and objectives may lead to better solutions, bypassing several local minima.

6.2 Publications from this thesis

a. Journal Publications

(1) A. Ch. Kapoutsis, S. A. Chatzichristofis, E. B. Kosmatopoulos: "A distributed, plug-n-play algorithm for multi-robot applications with a priori non-computable objective functions", The International Journal of Robotics Research, *submitted for publication.*

(2) A. Ch. Kapoutsis, S. A. Chatzichristofis, E. B. Kosmatopoulos, "DARP: Divide Areas Algorithm for Optimal Multi-Robot Coverage Path Planning", Journal of Intelligent & Robotic Systems, Springer, Volume 86, Issue 3, June 2017, pp 663-680.

(3) A. Ch. Kapoutsis, S. A. Chatzichristofis, L. Doitsidis, E. Kosmatopoulos, J. Sousa, J. Pinto and J. Braga, "Real-Time Adaptive Multi-Robot Exploration with Application to Underwater Map Construction", Autonomous Robots, Springer, Volume 40, Issue 6, August 2016, pp 987-1015.

b. Conference Proceedings

(1) A. Ch. Kapoutsis, G. Salavasidis, S. A. Chatzichristofis, J. Braga, J. Pinto, J. B. Sousa, Elias B. Kosmatopoulos, "The NOPTILUS Project Overview: A Fully-Autonomous Navigation System of Teams of AUVs for Static/Dynamic Underwater Map Construction", IFAC Workshop on Navigation, Guidance and Control of Underwater Vehicles (NGCUV'2015), April 28-30 2015, Girona - Catalonia (Spain), 2015.

(2) A. Ch. Kapoutsis, S. A. Chatzichristofis, L. Doitsidis, J. Borges de Sousa and E. B. Kosmatopoulos, "Autonomous Navigation of Teams of Unmanned Aerial or Underwater Vehicles for Exploration of Unknown Static & Dynamic Environments", 21st Mediterranean Conference on Control and Automation, (MED'13), pp. 1181-1188, June 25-28, 2013, Platanias-Chania, Crete, Greece.

(3) S. A. Chatzichristofis, A. Ch. Kapoutsis, E. B. Kosmatopoulos, L. Doitsidis, D. Rovas and Joao Borges de Sousa, "The NOPTILUS Project: Autonomous Multi-AUV Navigation for Exploration of Unknown Environments", FAC Workshop on Navigation, Guidance and Control of Underwater Vehicles (NGCUV'2012), April 10-12, 2012, Porto, Portugal.

c. Book Chapters

(1) A. Ch. Kapoutsis, S. A. Chatzichristofis, G. Ch. Sirakoulis, L. Doitsidis, and E. B. Kosmatopoulos, "Employing Cellular Au-

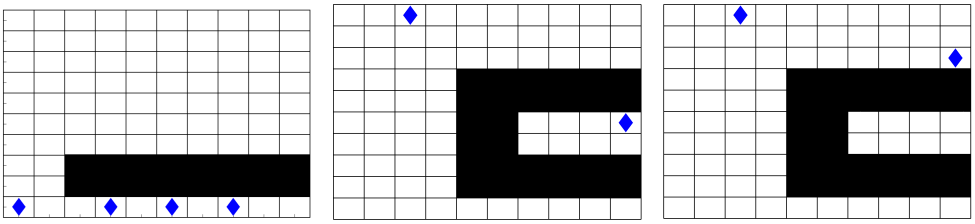
tomata for Shaping Accurate Morphology Maps Using Scattered Data from Robotics' Missions", Robots and Lattice Automata, Emergence, Complexity and Computation Volume 13, Springer-Verlag, G. Sirakoulis, A. Aadamatzky (Eds), 2015, pp 229-246.

A

DARP - Set-ups where an optimal space division does not exist

The problem formulation, as it is defined in section 3.2, 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 cannot be attained, due to the initial placements of the robots (sub-figure A.1(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



(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

Figure A.1. *Cases where the robots and/or obstacles arrangement, do not allow the acquisition of the optimal solution*

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 A.1(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).

Moreover, there are non-recoverable cases, where one or more sub-areas cannot be reached (sub-figure A.1(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.

B Multi-robot exploration based on the EKF error covariance matrix

The last few years, special attention have been paid in developing techniques for *active exploration* (active SLAM), see e.g., [30, 125, 126] and the references therein: using the information received so far, the robot next positions are designed so they optimize the mapping information of the SLAM algorithm. One possible way to attack such a problem is as follows: check all feasible next robot positions (e.g., all next robot positions that do not violate obstacle avoidance, maximum speed, communication, etc constraints) and find the ones that optimize some information metric that corresponds to the accuracy of the SLAM algorithm; then, move to the positions that optimize this information metric, and so on. Different types of such information metrics have been proposed, with the most popular being the trace of the EKF error covariance matrix, see e.g., [125, 126]. In such a case the robots are moving to the next positions that minimize the average (expected) EKF estimation error.

There two big issues with the above mentioned approach: the first is scalability, since it is computationally not feasible to check all possible combinations of next robots positions (this is practically infeasible even in the single robot case). There are, of course, many different approaches that relax the computational requirement of checking all possible next positions at the expense of sacrificing efficiency. However, even in the unrealistic case where infinite computing power would be available, as these algorithms are based on EKF – which, in turn, is based on linearizing the nonlinear multi-robot/sensor dynamics – the presence of nonlinear constraints (e.g., for obstacle avoidance or for not leaving a pre-specified area) may be destructive to the efficiency of the overall active exploration mission. The results of such a case are depicted in Figure B.1: three robots

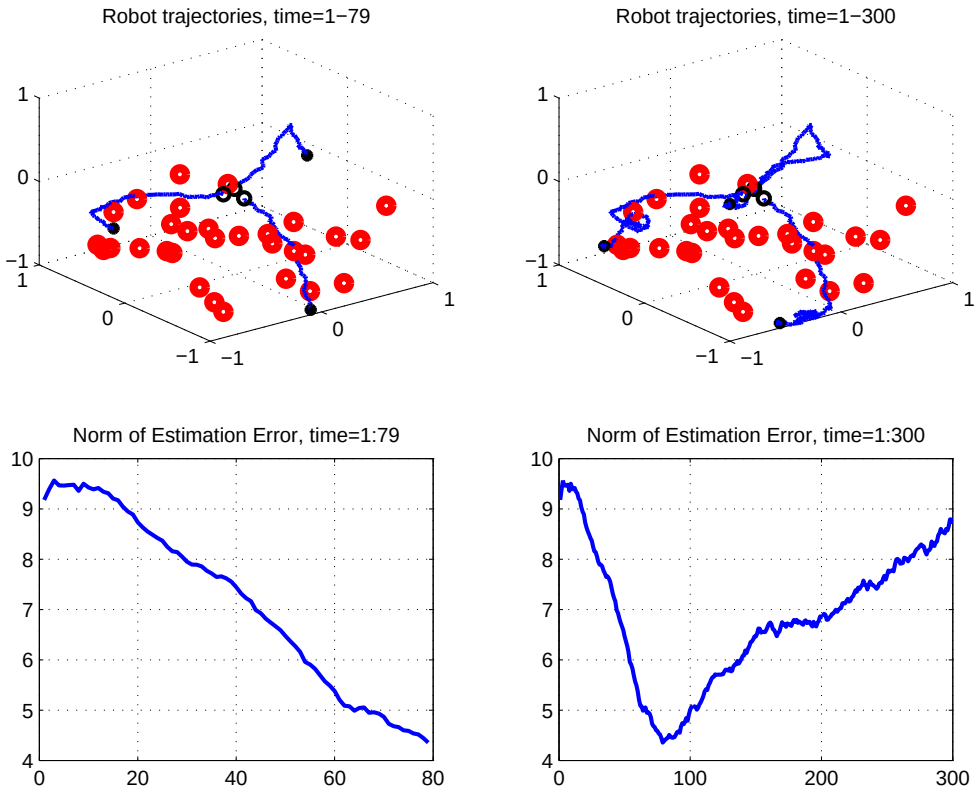


Figure B.1. *Autonomous exploration by moving towards minimizing the trace of EKF error covariance matrix: 3 robots, 30 landmarks, by assuming unlimited visibility, perfect localization and infinite computing power. The estimation error starts diverging as soon as the robots hit the boundary of the cube* $[-1,+1]^3$ *the robots are constrained to remain within.*

have been deployed for estimating the location of 30 static landmarks and their trajectories are designed so they minimize the trace of the EKF error covariance matrix, while they avoid obstacles (landmarks) and they remain within the cube $[-1,+1]^3$. Although, in the time-interval $[0,79]$ the overall algorithm behaves quite efficiently, it starts diverging as soon as the robots "hit" the boundaries of the area they have to remain within.

Bibliography

[1] 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. (pages 16, 28, 106, 115, 117, 119, and 120).

[2] Alessandro Renzaglia, Lefteris Doitsidis, Agostino Martinelli, and Elias B. Kosmatopoulos. Multi-robot three-dimensional coverage of unknown areas. *The International Journal of Robotics Research*, 31(6):738–752, 2012. (pages 16, 59, 73, 82, 86, 106, 119, 122, 124, 125, and 126).

[3] Ken Goldberg. Robotics: Countering singularity sensationalism. *Nature*, 526(7573):320–321, 2015. (page 21).

[4] Wenfeng Li, Junrong Bao, and Weiming Shen. Collaborative wireless sensor networks: A survey. In *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, pages 2614–2619. IEEE, 2011. (page 22).

[5] Shannon Hood, Kelly Benson, Patrick Hamod, Daniel Madison, Jason M O'Kane, and Ioannis Rekleitis. Bird's eye view: Cooperative exploration by ugv and uav. In *Unmanned Aircraft Systems (ICUAS), 2017 International Conference on*, pages 247–255. IEEE, 2017. (page 22).

[6] Sara Minaeian, Jian Liu, and Young-Jun Son. Vision-based target detection and localization via a team of cooperative uav and ugvs. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(7):1005–1016, 2016. (page 22).

[7] Jianqiang Li, Genqiang Deng, Chengwen Luo, Qiuzhen Lin, Qiao Yan, and Zhong Ming. A hybrid path planning method in unmanned

air/ground vehicle (uav/ugv) cooperative systems. *IEEE Transactions on Vehicular Technology*, 65(12):9585–9596, 2016. (page 22).

[8] Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12):1258–1276, 2013. (pages 25 and 37).

[9] 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. (pages 25 and 37).

[10] 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. (pages 25, 26, 34, and 38).

[11] 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. (page 26).

[12] 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. (pages 26 and 50).

[13] 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. (pages 26 and 37).

[14] 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. (page 27).

[15] 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. (page 27).

[16] Ariel Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica: Journal of the Econometric Society*, pages 97–109, 1982. (page 27).

[17] Hannah Bast and Susan Hert. The area partitioning problem, 2000. (page 27).

[18] 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. (pages 27 and 35).

[19] 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. (page 28).

[20] 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. (pages 28, 115, and 121).

[21] 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. (page 28).

[22] Stuart P Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982. (page 28).

[23] 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. (page 28).

[24] Franz Aurenhammer. Voronoi diagrams - a survey of a fundamental geometric data structure. *ACM Computing Surveys (CSUR)*, 23(3):345–405, 1991. (page 28).

[25] Jorge Cortés. Coverage optimization and spatial load balancing by robotic sensor networks. *Automatic Control, IEEE Transactions on*, 55(3):749–754, 2010. (page 28).

[26] Bruce Donald, Patrick Xavier, John Canny, and John Reif. Kinodynamic motion planning. *Journal of the ACM (JACM)*, 40(5):1048–1066, 1993. (page 28).

[27] Michail G Lagoudakis, Evangelos Markakis, David Kempe, Pinar Keskinocak, Anton J Kleywegt, Sven Koenig, Craig A Tovey, Adam Meyerson, and Sonal Jain. Auction-based multi-robot routing. In *Robotics: Science and Systems*, volume 5, page 343C350. Rome, Italy, 2005. (page 28).

[28] Fabio Pasqualetti, Antonio Franchi, and Francesco Bullo. On cooperative patrolling: Optimal trajectories, complexity analysis, and approximation algorithms. *Robotics, IEEE Transactions on*, 28(3):592–606, 2012. (page 28).

[29] Georg S Seyboth, Dimos V Dimarogonas, Karl Henrik Johansson, Paolo Frasca, and Frank Allgöwer. On robust synchronization of heterogeneous linear multi-agent systems with static couplings. *Automatica*, 53:392–399, 2015. (page 29).

[30] Jerome Le Ny and George J Pappas. On trajectory optimization for active sensing in gaussian process models. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 6286–6292. IEEE, 2009. (pages 29 and 149).

[31] Celso De La Cruz and Ricardo Carelli. Dynamic model based formation control and obstacle avoidance of multi-robot systems. *Robotica*, 26(03):345–356, 2008. (page 29).

[32] Laëtitia Matignon, Laurent Jeanpierre, and Abdel-Illah Mouaddib. Coordinated multi-robot exploration under communication constraints using decentralized markov decision processes. In *AAAI 2012*, pages p2017–2023, 2012. (page 29).

[33] Matthijs TJ Spaan and Nikos Vlassis. Perseus: Randomized point-based value iteration for pomdps. *Journal of artificial intelligence research*, 24:195–220, 2005. (page 29).

[34] Jesus Capitan, Matthijs T.J. Spaan, Luis Merino, and Anibal Ollero. Decentralized multi-robot cooperation with auctioned pomdps. *The International Journal of Robotics Research*, 32(6):650–671, 2013. (page 29).

[35] Nicholas Roy and Sebastian Thrun. Coastal navigation with mobile robots. In *NIPS*, volume 13, pages 1043–1049, 1999. (page 29).

[36] Zijian Wang and Mac Schwager. Multi-robot manipulation without communication. In *Distributed Autonomous Robotic Systems*, pages 135–149. Springer, 2016. (page 29).

[37] Ke Zhou and Stergios I Roumeliotis. Multirobot active target tracking with combinations of relative observations. *IEEE Transactions on Robotics*, 27(4):678–695, 2011. (page 29).

[38] Yu. Nesterov. Gradient methods for minimizing composite objective function. CORE Discussion Papers 2007076, Catholic University of Louvain, Center for Operations Research and Econometrics (CORE), 2007. (pages 29 and 30).

[39] Daniel Morgan, Giri P Subramanian, Soon-Jo Chung, and Fred Y Hadaegh. Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming. *The International Journal of Robotics Research*, 35(10):1261–1285, 2016. (pages 29 and 31).

[40] Jianing Chen, Melvin Gauci, Wei Li, Andreas Kolling, and Roderich Groß. Occlusion-based cooperative transport with a swarm of miniature mobile robots. *IEEE Transactions on Robotics*, 31(2):307–321, 2015. (page 29).

[41] Jorge Gomes, Paulo Urbano, and Anders Lyhne Christensen. Evolution of swarm robotics systems with novelty search. *Swarm Intelligence*, 7(2-3):115–144, 2013. (page 29).

[42] Martijn N Rooker and Andreas Birk. Multi-robot exploration under the constraints of wireless networking. *Control Engineering Practice*, 15(4):435–445, 2007. (pages 29, 59, and 60).

[43] Wolfram Burgard, Mark Moors, Cyrill Stachniss, and Frank E Schneider. Coordinated multi-robot exploration. *IEEE Transactions on robotics*, 21(3):376–386, 2005. (page 29).

[44] Cyrill Stachniss and Wolfram Burgard. Exploring unknown environments with mobile robots using coverage maps. In *IJCAI*, pages 1127–1134, 2003. (page 29).

[45] Rongxin Cui, Yang Li, and Weisheng Yan. Mutual information-based multi-auv path planning for scalar field sampling using multidimensional rrt. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(7):993–1004, 2016. (page 29).

[46] Frederic Bourgault, Alexei A Makarenko, Stefan B Williams, Ben Grocholsky, and Hugh F Durrant-Whyte. Information based adaptive robotic exploration. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 1, pages 540–545. IEEE, 2002. (page 29).

[47] John R Spletzer and Camillo J Taylor. Dynamic sensor planning and control for optimally tracking targets. *The International Journal of Robotics Research*, 22(1):7–20, 2003. (page 29).

[48] Randal W Beard, Timothy W McLain, Michael A Goodrich, and Erik P Anderson. Coordinated target assignment and intercept for unmanned air vehicles. *Robotics and Automation, IEEE Transactions on*, 18(6):911–922, 2002. (page 29).

[49] José Manuel Palacios-Gasós, Eduardo Montijano, Carlos Sagüés, and Sergio Llorente. Distributed coverage estimation and control for multirobot persistent tasks. *IEEE Transactions on Robotics*, 32(6):1444–1460, 2016. (pages 30, 31, 106, 133, 135, and 136).

[50] Ravi Kulan Rathnam and Andreas Birk. A distributed algorithm for cooperative 3d exploration under communication constraints. *Paladyn, Journal of Behavioral Robotics*, 4(4):223–232, 2013. (pages 30, 31, and 59).

[51] Athanasios Kapoutsis, Georgios Salavasidis, Savvas Chatzichristofis, Jose Braga, Jose Pinto, João Sousa, and Elias Kosmatopoulos. The noptilus project overview: A fully-autonomous navigation system of teams of auvs for static/dynamic underwater map construction. *IFAC-PapersOnLine*, 48(2):231–237, 2015. (page 30).

[52] Thomas Kollar and Nicholas Roy. Trajectory optimization using reinforcement learning for map exploration. *The International Journal of Robotics Research*, 27(2):175–196, 2008. (page 30).

[53] Nate Kohl and Peter Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Robotics and Automation,*

2004. Proceedings. ICRA'04. 2004 IEEE International Conference on, volume 3, pages 2619–2624. IEEE, 2004. (page 30).

[54] David Moloney and Oscar Deniz Suarez. A vision for the future [soapbox]. *Consumer Electronics Magazine, IEEE*, 4(2):40–45, 2015. (page 34).

[55] 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. (pages 34, 111, 114, and 119).

[56] Athanasios Kapoutsis, Savvas Chatzichristofis, Lefteris Doitsidis, Joao Borges de Sousa, and Elias B Kosmatopoulos. 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. (pages 34 and 111).

[57] Davide Scaramuzza, Michael C Achtelik, Lefteris Doitsidis, Fraundorfer Friedrich, Elias Kosmatopoulos, Agostino Martinelli, Markus W Achtelik, Margarita Chli, Savvas Chatzichristofis, Laurent Kneip, et al. Vision-controlled micro flying robots: from system design to autonomous navigation and mapping in gps-denied environments. *IEEE Robotics & Automation Magazine*, 21(3):26–40, 2014. (pages 34, 119, and 121).

[58] 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. (page 34).

[59] 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. (page 34).

[60] 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. (page 34).

[61] 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. (page 34).

[62] Howie Choset. Coverage for robotics–a survey of recent results. *Annals of mathematics and artificial intelligence*, 31(1-4):113–126, 2001. (page 34).

[63] 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. (page 34).

[64] 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. (page 34).

[65] 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. (page 34).

[66] Xiaoming Zheng, Sven Koenig, David Kempe, and Sonal Jain. Multirobot forest coverage for weighted and unweighted terrain. *IEEE Transactions on Robotics*, 26(6):1018–1031, 2010. (pages 35, 37, and 50).

[67] 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. (page 35).

[68] Stephen J Wright. Coordinate descent algorithms. *Mathematical Programming*, 151(1):3–34, 2015. (pages 36, 42, 104, and 114).

[69] Wesley H Huang. Optimal line-sweep-based decompositions for coverage algorithms. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 1, pages 27–32. IEEE, 2001. (page 37).

[70] Howie M Choset. *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005. (page 37).

[71] Robert Endre Tarjan. *Data structures and network algorithms*, volume 14. SIAM, 1983. (page 38).

[72] R Nandakumar and Ramana Rao. Fair partitions of polygons: An elementary introduction. *Proceedings-Mathematical Sciences*, 122(3):459–467, 2012. (page 40).

[73] Andreas Birk, Max Pfingsthorn, and Heiko Bülow. Advances in underwater mapping and their application potential for safety, security, and rescue robotics. In *IEEE International Symposium on Safety, Security, Rescue Robotics (SSRR)*. IEEE Press, IEEE Press, 2012. (page 56).

[74] Scott Reed, Jon Wood, and Chris Haworth. The detection and disposal of ied devices within harbor regions using auvs, smart rovs and data processing/fusion technology. In *Waterside Security Conference (WSS), 2010 International*, pages 1–7. IEEE, 2010. (page 56).

[75] Anders Rodningsby and Yaakov Bar-Shalom. Tracking of divers using a probabilistic data association filter with a bubble model. *IEEE Transactions on Aerospace and Electronic Systems*, 45(3), 2009. (page 56).

[76] Ronald T Kessel and Reginald D Hollett. Underwater intruder detection sonar for harbour protection: State of the art review and implications. Technical report, DTIC Document, 2006. (page 56).

[77] Robin R Murphy, Eric Steimle, Michael Hall, Michael Lindemuth, David Trejo, Stefan Hurlebaus, Zenon Medina-Cetina, and Daryl Slocum. Robot-assisted bridge inspection after hurricane ike. In *Safety, Security & Rescue Robotics (SSRR), 2009 IEEE International Workshop on*, pages 1–5. IEEE, 2009. (page 56).

[78] Kevin Kochersberger, Kenneth Kroeger, Bryan Krawiec, Eric Brewer, and Thomas Weber. Post-disaster remote sensing and sampling via an autonomous helicopter. *Journal of Field Robotics*, 31(4):510–521, 2014. (page 56).

[79] C. Roman and R. Mather. Autonomous underwater vehicles as tools for deep-submergence archaeology. *Proceedings of the Institution of*

Mechanical Engineers, Part M: Journal of Engineering for the the Maritime Environment, 224:327–340, 2010. (page 56).

[80] Matthew Johnson-Roberson, Mitch Bryson, Ariell Friedman, Oscar Pizarro, Giancarlo Troni, Paul Ozog, and Jon C Henderson. High-resolution underwater robotic vision-based mapping and three-dimensional reconstruction for archaeology. *Journal of Field Robotics*, 34(4):625–643, 2017. (page 56).

[81] James E DeVault. Robotic system for underwater inspection of bridge piers. *IEEE Instrumentation & Measurement Magazine*, 3(3):32–37, 2000. (page 56).

[82] Philippe Blondel. A review of acoustic techniques for habitat mapping. *Hydroacoustics*, 11:29–38, 2008. (page 56).

[83] Javed Khurshid and Hong Bing-Rong. Military robots-a glimpse from today and tomorrow. In *Control, Automation, Robotics and Vision Conference, 2004. ICARCV 2004 8th*, volume 1, pages 771–777. IEEE, 2004. (page 56).

[84] Abd Manan Samad, Nazrin Kamarulzaman, Muhammad Asyraf Hamdani, Thuaibatul Aslamiah Mastor, and Khairil Afendy Hashim. The potential of unmanned aerial vehicle (uav) for civilian and mapping application. In *System Engineering and Technology (ICSET), 2013 IEEE 3rd International Conference on*, pages 313–318. IEEE, 2013. (page 56).

[85] George Adamides, Christos Katsanos, Ioannis Constantinou, Georgios Christou, Michalis Xenos, Thanasis Hadzilacos, and Yael Edan. Design and development of a semi-autonomous agricultural vineyard sprayer: Human–robot interaction aspects. *Journal of Field Robotics*, 34(8):1407–1426, 2017. (page 56).

[86] MWM Gamini Dissanayake, Paul Newman, Steve Clark, Hugh F Durrant-Whyte, and Michael Csorba. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on robotics and automation*, 17(3):229–241, 2001. (page 56).

[87] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann. 6d slam - 3d mapping outdoor environments. *Journal of Field Robotics*, 24(8-9):699–722, 2007. (page 56).

[88] Sebastian Thrun and Michael Montemerlo. The graph slam algorithm with applications to large-scale mapping of urban structures. *The International Journal of Robotics Research*, 25(5-6):403–429, 2006. (page 56).

[89] Elias B Kosmatopoulos, Markos Papageorgiou, Antigoni Vakouli, and Anastasios Kouvelas. Adaptive fine-tuning of nonlinear control systems with application to the urban traffic control strategy tuc. *IEEE Transactions on Control Systems Technology*, 15(6):991–1002, 2007. (pages 59, 73, and 82).

[90] Elias B Kosmatopoulos. An adaptive optimization scheme with satisfactory transient performance. *Automatica*, 45(3):716–723, 2009. (pages 59, 73, 82, 84, 85, 86, 112, and 113).

[91] Elias B Kosmatopoulos and Anastasios Kouvelas. Large scale nonlinear control system fine-tuning through learning. *IEEE Transactions on Neural Networks*, 20(6):1009–1023, 2009. (pages 59, 73, 82, 85, 86, 112, and 113).

[92] Dieter Fox, Jonathan Ko, Kurt Konolige, Benson Limketkai, Dirk Schulz, and Benjamin Stewart. Distributed multirobot exploration and mapping. *Proceedings of the IEEE*, 94(7):1325–1339, 2006. (page 59).

[93] Julian De Hoog, Stephen Cameron, and Arnoud Visser. Role-based autonomous multi-robot exploration. In *Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, 2009. COMPUTATIONWORLD'09. Computation World:*, pages 482–487. IEEE, 2009. (page 59).

[94] Luigi Freda and Giuseppe Oriolo. Frontier-based probabilistic strategies for sensor-based exploration. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 3881–3887. IEEE, 2005. (page 59).

[95] Wolfram Burgard, Mark Moors, Dieter Fox, Reid Simmons, and Sebastian Thrun. Collaborative multi-robot exploration. In *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, volume 1, pages 476–481. IEEE, 2000. (page 59).

[96] Brian Yamauchi. Frontier-based exploration using multiple robots. In *Proceedings of the second international conference on Autonomous agents*, pages 47–53. ACM, 1998. (page 59).

[97] Max Pfingsthorn, Andreas Birk, and H Bulow. An efficient strategy for data exchange in multi-robot mapping under underwater communication constraints. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 4886–4893. IEEE, 2010. (page 60).

[98] Benjamin Johnson, Nicodemus Hallin, Hans Leidenfrost, Michael O'Rourke, and Dean Edwards. Collaborative mapping with autonomous underwater vehicles in low-bandwidth conditions. In *OCEANS 2009-EUROPE*, pages 1–7. IEEE, 2009. (page 60).

[99] Andrew Rajala and Dean Edwards. *Allocating auvs for mine map development in mcm*. IEEE, 2007. (page 60).

[100] Lefteris Doitsidis, Stephan Weiss, Alessandro Renzaglia, Markus W Achtelik, Elias Kosmatopoulos, Roland Siegwart, and Davide Scaramuzza. Optimal surveillance coverage for teams of micro aerial vehicles in gps-denied environments using onboard vision. *Autonomous Robots*, 33(1-2):173–188, 2012. (pages 73 and 82).

[101] Alessandro Renzaglia, Lefteris. Doitsidis, Savvas Chatzichristofis, Agostino Martinelli, and Elias Kosmatopoulos. Distributed multi-robot coverage using micro aerial vehicles. In *21st Mediterranean Conference on Control and Automation, (MED13)*, pages 963–968, Chania, Greece, 2013. (page 73).

[102] Simone Baldi, Iakovos Michailidis, Elias B Kosmatopoulos, and Petros A Ioannou. A "plug and play" computationally efficient approach for control design of large-scale nonlinear systems using cosimulation: a combination of two "ingredients". *IEEE Control Systems*, 34(5):56–71, 2014. (page 75).

[103] A. Amanatiadis, S. A. Chatzichristofis, K. Charalampous, L. Doitsidis, E. B. Kosmatopoulos, P. Tsalides, A. Gasteratos, and S. Roumeliotis. A multi-objective exploration strategy for mobile robots under operational constraints. *IEEE Access*, 1:691–702, 2013. (pages 82 and 114).

[104] David Ruppert and Matthew P Wand. Multivariate locally weighted least squares regression. *The annals of statistics*, pages 1346–1370, 1994. (page 87).

[105] Gianluca Antonelli, Filippo Arrichiello, Fabrizio Caccavale, and Alessandro Marino. Decentralized time-varying formation control for multi-robot systems. *The International Journal of Robotics Research*, page 0278364913519149, 2014. (pages 106, 130, 131, and 133).

[106] Marios M Polycarpou and Petros A Ioannou. *Identification and control of nonlinear systems using neural network models: Design and stability analysis*. University of Southern Calif., 1991. (page 110).

[107] James W Demmel. *Applied numerical linear algebra*. SIAM, 1997. (page 112).

[108] Dimitri P Bertsekas. *Nonlinear programming*. Athena scientific Belmont, 1999. (pages 113 and 114).

[109] 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. (page 113).

[110] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU Press, 2012. (page 114).

[111] Christos D Korkas, Simone Baldi, Iakovos Michailidis, and Elias B Kosmatopoulos. Occupancy-based demand response and thermal comfort optimization in microgrids with renewable energy sources and energy storage. *Applied Energy*, 163:93–104, 2016. (page 114).

[112] Mac Schwager, James McLurkin, and Daniela Rus. Distributed coverage control with sensory feedback for networked robots. In *robotics: science and systems*, Philadelphia, USA, 2006. (page 115).

[113] Efrat Sless, Noa Agmon, and Sarit Kraus. Multi-robot adversarial patrolling: facing coordinated attacks. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 1093–1100. International Foundation for Autonomous Agents and Multiagent Systems, 2014. (page 115).

[114] LE Caraballo, JJ Acevedo, JM Díaz-Báñez, BC Arrue, I Maza, and A Ollero. The block-sharing strategy for area monitoring missions using a decentralized multi-uav system. In *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, pages 602–610. IEEE, 2014. (page 115).

[115] Joaquin López, Diego Pérez, Enrique Paz, and Alejandro Santana. Watchbot: A building maintenance and surveillance system based on autonomous robots. *Robotics and Autonomous Systems*, 61(12):1559–1571, 2013. (page 115).

[116] Alberto Quattrini Li, Riccardo Cipolleschi, Michele Giusto, and Francesco Amigoni. A semantically-informed multirobot system for exploration of relevant areas in search and rescue settings. *Autonomous Robots*, 40(4):581–597, 2016. (page 115).

[117] Zoltán Beck, Luke Teacy, Alex Rogers, and Nicholas R Jennings. Online planning for collaborative search and rescue by heterogeneous robot teams. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 1024–1033. International Foundation for Autonomous Agents and Multiagent Systems, 2016. (page 115).

[118] Georgios Salavasidis, Catherine Harris, Stephen McPhail, Alexander B Phillips, and Eric Rogers. Terrain aided navigation for long range auv operations at arctic latitudes. In *Autonomous Underwater Vehicles (AUV), 2016 IEEE/OES*, pages 115–123. IEEE, 2016. (page 121).

[119] Francisco Curado Teixeira. Terrain-aided navigation and geophysical navigation of autonomous underwater vehicles. *Instituto Superior Técnico, Lisboa*, 2007. (page 121).

[120] Lefteris Doitsidis, Stephan Weiss, Alessandro Renzaglia, Markus W Achtelik, Elias Kosmatopoulos, Roland Siegwart, and Davide Scaramuzza. Optimal surveillance coverage for teams of micro aerial vehicles in gps-denied environments using onboard vision. *Autonomous Robots*, 33(1-2):173–188, 2012. (page 122).

[121] Matthias Nieuwenhuisen, David Droeschel, Marius Beul, and Sven Behnke. Obstacle detection and navigation planning for autonomous micro aerial vehicles. In *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, pages 1040–1047. IEEE, 2014. (page 134).

[122] Nisheeth Shrivastava, R Mudumbai U Madhow, and S Suri. Target tracking with binary proximity sensors: fundamental limits, minimal descriptions, and algorithms. In *Proceedings of the 4th international*

conference on Embedded networked sensor systems, pages 251–264. ACM, 2006. (page 134).

[123] Wooyoung Kim, Kirill Mechitov, J-Y Choi, and Soo Ham. On target tracking with binary proximity sensors. In *Information Processing in Sensor Networks, 2005. IPSN 2005. Fourth International Symposium on*, pages 301–308. IEEE, 2005. (page 134).

[124] Fred Y Hadaegh, Soon-Jo Chung, and Harish M Manohara. On development of 100-gram-class spacecraft for swarm applications. *IEEE Systems Journal*, 10(2):673–684, 2016. (page 139).

[125] Faraz M. Mirzaei, Anastasios I. Mourikis and Stergios I. Roumeliotis. Analysis of positioning uncertainty in cooperative localization and target tracking (CLATT). Technical report, Dept. of Computer Science & Engineering, University of Minnesota Minneapolis, MN 55455, 2005. (page 149).

[126] Robert Sim and Nicholas Roy. Global a-optimal robot exploration in slam. In *ICRA*, pages 661–666, 2005. (page 149).