Adaptive stochastic optimization for cooperative coverage with a swarm of Micro Aerial Vehicles

Alessandro Renzaglia

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Mathématiques-informatique**

Arrêté ministérial : 7 août 2006

Présentée par

Alessandro RENZAGLIA

Thèse dirigée par **Thierry FRAICHARD**
et codirigée par **Agostino MARTINELLI**

préparée au sein du centre de recherche **INRIA Rhône-Alpes**,
du **Laboratoire d'Informatique de Grenoble**
dans l'Ecole Doctorale **Mathématiques, Sciences et Technologies de l'Information, Informatique**

Adaptive stochastic optimization for cooperative coverage with a swarm of Micro Aerial Vehicles

Thèse soutenue publiquement le **27 Avril 2012**,
devant le jury composé de :

Elias KOSMATOPOULOS
Professeur DUTH/CERTH, Thessalonique, Grèce, Président

Simon LACROIX
Directeur de Recherche LAAS/CNRS, Toulouse, France, Rapporteur

Xiaoming HU
Professeur KTH, Stockholm, Suède, Rapporteur

Thierry FRAICHARD
Chargé de Recherche INRIA Rhône-Alpes, Grenoble, France, Directeur de thèse

Agostino MARTINELLI
Chargé de Recherche INRIA Rhône-Alpes, Grenoble, France, Co-Directeur de thèse

Acknowledgments

This thesis has been a great and completely positive experience not only from a professional point of view, but also human and for that I want to thank several people. First of all my advisor, and by now friend, Agostino Martinelli. He firstly gave me the opportunity to start this thesis, and then he has always been available to teach, discuss, motivate and not least, share great experiences on the mountains: in other words he has been veramente massiccio!!

I am also very grateful to Elias Kosmatopoulos, from whom I learnt a lot, for his great support and motivation during these years. Moreover, the wonderful hospitality during the time I spent in Greece was incomparable. And of course also to Lefteris Doitsidis: working with them has been a very good opportunity. For my time in Greece I also have to thank Savvas Chatzichristofis and all the other guys in Xanthi: it was a short but very nice time and I felt immediately one of them.

Then I would thank the reviewers, Simon Lacroix and Xiaoming Hu, for their important and constructive comments, Thierry Fraichard for his advices, all the members of the e-Motion team to make the work environment so pleasant, and Myriam for her very important administrative help.

My thesis was carried out in the framework of the European project sFly. The support of this project was not just financial: it allowed me to collaborate with excellent researchers and certainly I have been enriched by this experience. So thanks to all the project members and in particular to Davide Scaramuzza, a perfect project coordinator.

Of course, these years were much more than just this thesis and I want to thank all my friends in Grenoble.

Un ringraziamento speciale va ovviamente alla mia famiglia che mi ha sempre supportato, anche se in questi anni a distanza, rendendo tutto più facile. Grazie di tutto!

Infine grazie a Lara, che ha dato a questa tesi tutta un'altra importanza!

Abstract

The use of multi-robot teams has gained a lot of attention in recent years. This is due to the extended capabilities that the teams offer compared to the use of a single robot for the same task. Moreover, as these platforms become more and more affordable and robust, the use of teams of aerial vehicles is becoming a viable alternative. This thesis focuses on the problem of deploying a swarm of Micro Aerial Vehicles (MAV) to perform surveillance coverage missions over an unknown terrain of arbitrary morphology. Since the terrain's morphology is unknown and it can be quite complex and non-convex, standard algorithms are not applicable to the particular problem treated in this thesis. To overcome this, a new approach based on the Cognitive-based Adaptive Optimization (CAO) algorithm is proposed and evaluated. A fundamental property of this approach is that it shares the same convergence characteristics as those of constrained gradient-descent algorithms, which require perfect knowledge of the terrain's morphology to optimize coverage. In addition, it is also proposed a different formulation of the problem in order to obtain a distributed solution, which allows us to overcome the drawbacks of a centralized approach and to consider also limited communication capabilities. Rigorous mathematical arguments and extensive simulations establish that the proposed approach provides a scalable and efficient methodology that incorporates any particular physical constraints and limitations able to navigate the robots to an arrangement that (locally) optimizes the surveillance coverage. The proposed method is finally implemented in a real swarm of MAVs to carry out surveillance coverage in an outdoor complex area.

L'utilisation d'équipes de robots a pris de l'ampleur ces derniéres années. Cela est dû aux avantages que peut offrir une équipe de robot par rapport à un robot seul pour la réalisation d'une même tâche. Cela s'explique aussi par le fait que ce type de plates-formes deviennent de plus en plus abordables et fiables. Ainsi, l'utilisation d'une équipe de véhicules aériens devient une alternative viable. Cette thèse se concentre sur le problème du déploiement d'une équipe de Micro-Véhicules Aériens (MAV) pour effectuer des missions de surveillance sur un terrain inconnu de morphologie arbitraire. Puisque la morphologie du terrain est inconnue et peut être complexe et non-convexe, les algorithmes standards ne sont pas applicables au problème particulier traité dans cette thèse. Pour y remédier, une nouvelle approche basée sur un algorithme d'optimisation cognitive et adaptatif (CAO) est proposée et évaluée. Une propriété fondamentale de cette approche est qu'elle partage les mêmes caractéristiques de convergence que les algorithmes de descente de gradient avec contraintes qui exigent une connaissance parfaite de la morphologie du terrain pour optimiser la couverture. Il est également proposé une formulation différente du problème afin d'obtenir une solution distribuée, ce qui nous permet de surmonter les inconvénients d'une approche centralisée et d'envisager également des capacités de communication limitées. De rigoureux arguments mathématiques et des simulations étendues établissent que l'approche proposée fournit une méthodologie évolutive et efficace qui intègre toutes les contraintes physiques particulières et est capable de guider les robots vers un arrangement qui optimise localement la surveillance. Finalement, la méthode proposée est mise en œuvre sur une équipe de MAV réels pour réaliser la surveillance d'un environnement extérieur complexe.

Contents

List of Figures

Chapter 1

Introduction

The use of Unmanned Aerial Vehicles (UAVs) teams has gained a lot of attention in recent years. This is due to the extended capabilities that flying robots are able to offer comparing to the use of ground robots for the same task. The ability to fly allows easily avoiding obstacles on the ground and to have an excellent birds eye view. Moreover, it is possible to access to environments where no human or other vehicles can access to. Therefore flying robots are the logical heir of ground based mobile robots. If they are further realized in small scale, they can also be used in narrow out- and indoor environment and they represent only a limited risk for the environment and people living in it. Micro Aerial Vehicles (MAVs) teams can be used in a variety of very important missions including:

- Surveillance of buildings and large in- and outdoor areas: instead of fix mounted security cameras the micro-helicopters would allow a re-configurable grid of surveillance cameras and establish one when needed in places where security cameras do not exist.

- Rescue missions: aerial robots capable of flying in closed quarters and collapsed buildings could quickly and systematically search to locate victims of an accident or a natural disaster.

- Surveillance of dangerous areas, chemical and nuclear plants: the micro helicopters could explore areas that are dangerous to human personal, i.e. areas of chemical, biological or nuclear contamination.

- Environmental monitoring: the flying micro-robots would be an excellent tool for environmental monitoring (air quality, forest fire, ...) as individual unit, in a swarm or in connection with a sensor network.

Figure 1.1: Typical scenarios where the surveillance coverage is a fundamental task.

- Law enforcement in public area: a micro-helicopter could provide real time imagery to aid police during surveillance missions or criminal search operations.

In all the aforementioned tasks the deployment of limited resources (robots) to optimize the monitoring of the area of interest is the key issue. Moreover, as these platforms become more and more affordable and robust, the use of teams of aerial vehicles that cooperatively and autonomously search and cover an assigned area is becoming a viable alternative. In order to exploit the advantages of robot mobility, active sensing strategies need to be determined for coordinating the motion of groups of robots while optimizing the use of the available sensing, communication, and processing resources.

Furthermore, in every multi-robot systems, a distributed approach is desirable for several fundamental reasons. The most important are failure of the central station and limited communication capabilities. In a very common scenario each robot has no global knowledge about the surrounding environment or about the group as whole. So, the global behavior of the team can be seen as the sum of the local actions taken by its members, which sense their immediate environment, communicate with their neighbors, process the information gathered and move according to it.

1.1 Context of the thesis

The work of this thesis has been carried out in the framework of the European project sFly (www.sfly.org). The objective of this project is to develop several small and safe helicopters which can fly autonomously in city-like environments and which can be used to assist humans in tasks like rescue and monitoring. The main motivations of this work are not only to achieve tasks impossible for a human team, but also to be able to substitute the human intervention in very dangerous scenarios. This means that the helicopters must be able to operate in complex environments where GPS signals are often shadowed, cooperatively and in a complete

Figure 1.2: ATG delivered the 3 new helicopters version 3 (hexacopters), each one equipped with a Core2Duo computer board, camera mounts, and communication modules (WiseNodes) by the industrial partner CSEM.

autonomous way. This involves a number of challenges on all levels of helicopter design, perception, actuation, control, navigation and power supply that have yet to be solved. The visual-based navigation problematics have been addressed by the Autonomous System Lab (ASL) and the Computer Vision and Geometry Group (CVG) laboratories at ETH of Zurich, Switzerland. The helicopters have been appropriately developed for the project by Ascending Technologies GMBH (ATG), from Germany (see Fig. 1.2). The wireless communication and range measurement issues are considered by CSEM, Neuchatel, Switzerland. Finally, INRIA (Grenoble, France) and CERTH/TUC (Thessaloniki/Chania, Greece) have been the partners responsible for the active navigation guidance for cooperative tasks, the topic of this thesis.

1.2 Cooperative coverage

The problem of deploying a team of flying robots to perform surveillance coverage missions over an unknown terrain of complex and non-convex morphology is considered. This problem can be expressed as an optimization problem: given an arbitrary initial team configuration finding the final robots' positions which maximize the degree of coverage and the way to reach such a configuration. In this thesis we assume that the team surveillance capabilities are enough to achieve a satisfactory level of monitoring from a static configuration. In other words, we do not consider here the case of an environment too large to be monitored and which requires dynamical surveillance strategies. To quantify the degree of coverage in a

typical coverage mission two different main criteria may be identified:

(O1) the part of the terrain that is monitored (i.e., is visible) by the robot team has to be maximized;

(O2) for every point in the terrain, the closest robot has to be as close as possible to that point.

The first objective is the most intuitive in a surveillance task: finding the positions from which it is possible to see as more as possible, regarding the sensors capabilities of the team. In the thesis, we will refer to this problem as the visibility problem.

The second objective might be necessary for two practical reasons: (a) firstly, in many multi-robot coverage applications there is the necessity of being able to intervene as fast as possible in any of the points of the terrain with at least one robot and (b) secondly, the closer is the robot to a point in the terrain the better is, in general, its sensing ability to monitor this point. We will refer to this problem as the intervention problem.

Of course, finding the optimal positions for the robots team is not the unique problem to solve. Indeed, in many situations the optimization problem has to be solved on-line, starting from completely arbitrary positions, with no, or partial, a priori knowledge about the environment to monitor. So in this case also creating in real time safe trajectories, respecting all the physical and environmental constraints, is a further challenge.

Our goal is to develop an efficient and adaptive strategy to lead the robots to maximize the part of the terrain that is visible while keeping the distance between each point in the terrain and the closest team member as small as possible. A compromise between these two objectives should be fulfilled given the physical constraints and limitations imposed at the particular application. As the terrain morphology is unknown and it can be very complex and non-convex, standard algorithms are not applicable to the particular problem treated in this thesis. To overcome this, a new approach based on the Cognitive-based Adaptive Optimization (CAO) algorithm is proposed and evaluated. The CAO algorithm is a stochastic adaptive optimization method recently proposed by Kosmatopoulos in [61], [63]. In its first version, this method did not include the possibility to cope with constrained problems. A contribution of this thesis is to extend the CAO algorithm to overcome this limitation. A fundamental property of this approach is that it shares the same convergence characteristics as those of constrained gradient-descent algorithms, which require perfect knowledge of the terrain's morphology. Rigorous mathematical arguments and extensive simulations establish that the proposed approach provides a scalable and efficient methodology that incorporates any particular physical constraints and limitations able to navigate the robots to an arrangement that (locally) optimizes the surveillance coverage.

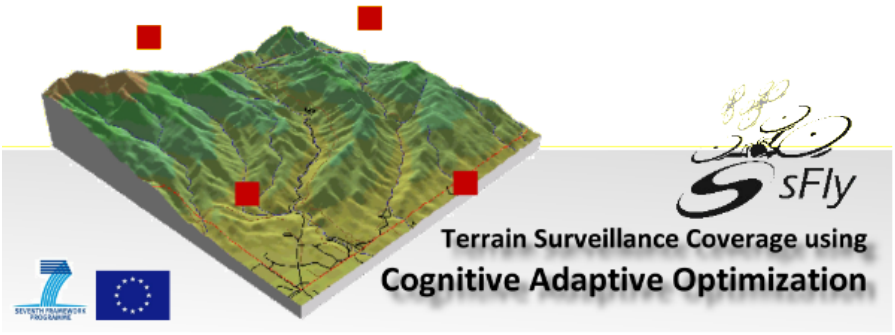


Figure 1.3: The work developed in this thesis is in the framework of the European project sFly.

1.2.1 Related Works

The coverage problem was defined in literature for the first time by Gage [31]. In this work, where the main application is for defense in a military scenario, the author introduces three different kinds of coverage: blanket coverage, barrier coverage and sweep coverage. In the first one the objective is an optimal static deployment that maximizes the total detection area; the barrier coverage has the objective to minimize the probability of undetected penetrations through a barrier that the robots form in defense of a given region; the last one, the sweep coverage, is a barrier coverage with a moving barrier. Following these definitions, the coverage considered in this thesis is a blanket coverage. As already said, we can divide the static coverage into two main problems: the intervention and the visibility problems.

Intervention problem

The majority of approaches for multi-robot static surveillance coverage concentrate on objective (O2) previously described, the intervention problem. The first work on this problem is of Cortés *et al.* [20]. Here the authors present a gradient-descent algorithm for the coverage of a convex region, i.e., without obstacles, with a team of mobile robots. This solution is based on the concept of centroidal Voronoi partition and adopts the Lloyd algorithm, [73], to lead the robots to the final positions. After this work, a great number of similar solutions have been proposed trying to adapt, extend or improve the original algorithm for a variety of different scenarios. One of them, always for a convex environment, is proposed by Schwager *et al.* in [105], where additionally the robots estimate a function indicating the relative importance of different areas in the environment, using information from the sensors. A possible extension of the previous work has been proposed in [74], where a solution for nonholonomic mobile

sensors is provided. Poduri and Sukhatme used the artificial potential field method to obtain a coverage of a convex region with the constraint that each robot has at least K neighbors [86]. In [39] a different definition of the Voronoi partition is proposed in order to consider an anisotropic sensor model, where the performance of the sensors depends not only on the distance but also on the orientation with respect to the target. Another variation on the classical Voronoi coverage control, inspired by the hunting tactics of ladybugs, is presented in [103]. By using this controller the final positions are always the centroidal configuration, but there is a greater degree of exploration carried out by the team during the task.

The same kind of coverage problem in a non-convex environment is more complex but also more interesting for practical applications. A possible solution to this problem is proposed by Pimenta *et al.* in [85]. Also in this case, the solution is based on Voronoi partition, but it is obtained using the geodesic distance instead of the Euclidean one. This choice allows taking into account the particular geometry of the environment. An extension of the previous work to entropy-based metrics, in order to allow for coverage in unknown non-convex environment, is presented in [10]. In [46], the same problem is approached by using the artificial potential field method: each robot feels a repulsive force from the obstacle and from the other robots. In this way the algorithm assures at the same time the spreading out of the team and the collision avoidance during the mission. Another possible solution for environments which include obstacles is proposed in [16]: the main idea is to combine the classical Voronoi coverage with the Lloyd algorithm and the local path planning algorithm TangentBug [52]. In all the aforementioned works the regions to cover are in 2D. In [17] the authors propose a solution for a particular class of non-convex regions based on diffeomorphic transformations to map the non-convex region to a convex one and then to use the classical Voronoi-based control law. Durham *et al.* tackle this problem in [25], tacking into account also asynchronous communication between each robot and a base station and with the rest of the team. In [15] the authors approach also the problem of deploying a team of robots on a non-planar surface in 3D space.

Visibility problem

As far as it concerns objective (O1) described in the previous section, the visibility problem, different solutions have been proposed in the literature. In [6], Batalin and Sukhatme present two methods based on a local dispersive interaction between robots to achieve good global coverage. In this case the problem is not approached defining an objective function to optimize and the solution is based only on a mutually dispersive interaction between robots when there is an overlapping of sensing region. In [32] the authors propose a gradient-based

algorithm for the case of a single robot case and they prove that the visible area is almost everywhere a locally Lipschitz function of the observer location. In [34], an approach for the multi-robot problem is presented based on the assumption that the environment is simply connected. The visibility problem is also related to the Art Gallery Problem where the goal is to find the optimum number of guards in a non-convex environment so that each point of the environment is visible by at least one guard, see [84], [117], [1], [106], [33] and references therein. All the aforementioned solutions are based on the hypothesis that a given point can be monitored regardless of its distance from the robot. An incremental algorithm which takes into consideration also a maximum monitoring distance is presented in [45]. In [104], the authors consider the coverage of a 2D region by using a team of hovering robots. In this case, information per pixel is proposed as optimization criterion. Laventall and Cortés, in [70], propose a Voronoi-based solution for the problem of the coverage of a convex environment with limited-range anisotropic sensors. Always using mobile anisotropic sensor networks, Hexsel *et al.* propose in [43] a distributed gradient-ascent algorithm to maximize the probability of detection of events in a 2D area with also polygonal obstacles.

Dynamic coverage

A different problem, always known as coverage (or complete coverage) and very studied in literature, is the determination of the path that a robot must take in order to pass over each point in an environment. This kind of coverage is more suitable for a searching, demining or cleaning mission or for map generation than for surveillance of an area. A typical strategy is to divide the region in non overlapping cells and then finding the sequence to cover all the cells. Choset and Pignon propose in [19] a solution based on this idea, while Gabriely and Rimon present in [30] a grid-based method to solve this problem. Also in [49], contrary to the coverage considered in this thesis, the goal is not to find a static final configuration but to dynamically cover a given region: once a predefined attained effective coverage is achieved, the mission is accomplished regardless to the positions of the agents. A similar, but cooperative, problem appears in [2], where the algorithm has been implemented on a team of UAVs equipped with fixed cameras.

The dynamic patrolling problem is another problem related to the surveillance coverage. The primary goal of many patrolling tasks is the capture of unknown entities within the environment. Usually, the difference with respect to the coverage problems is that instead of deploying a team of mobile robots, the starting point is a set of fixed cameras with the possibility to move their directions. Example in the literature can be found in [118], [5], [89]. A very close problem has been approached by Michael *et al.* in [79]. In this work the objective

is to monitor a building with discrete locations of interest by using a team of MAVs. Each of the locations must be observed periodically and the solution take into account also robots' power constraints.

To the best of our knowledge, the problem of considering the two objectives simultaneously to statically cover a completely arbitrary 3D region by using a team of flying robots has never been investigated so far. To do that we propose to use a new stochastic optimization method, the CAO algorithm. The many advantages of using stochastic gradient descent algorithms, like the Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm, to approach a sensor-based deployment problem have already been highlighted in [80]. In this work, the authors proposed applications such as: coverage with heterogeneous sensors and source seeking with stochastic wireless connectivity constraints. The same deployment problem but with additionally end-to-end communication constraint is considered in [81].

1.3 Organization and Contributions of the Thesis

According with the motivation of the project, the challenge of this thesis was to develop an efficient and adaptive methodology to perform cooperative surveillance coverage in a generic, complex terrain by using a swarm of Micro Aerial Vehicles. The difficulties of this challenge are not only in solving the complex mathematical problem but also related to the strong constraints that the final real application requires: low computational complexity, lack of information about the area to cover, possible limited communication capabilities, and so on. To address these challenges we adopted a new stochastic adaptive optimization algorithm. Firstly we extended this approach to make it suitable for the particular considered problem. Then, all the steps, from the simulations in simple 2D regions up to the implementation on a real swarm of helicopters have been carried out. Additionally, also a distributed version of the algorithm, for a particular coverage criterion, has been developed allowing us to include communication constraints and to avoid all the problems that a centralized approach can generate. The rest of this thesis is organized as follows.

Chapter 2: Stochastic Optimization

A large number of practical problems in mobile robotics can be expressed as optimization problems and a good optimization strategy can be of crucial importance to achieve many complex tasks. The aim of this chapter is to provide a quick introduction on stochastic optimization and present a review of the most common and exploited algorithms in the literature. This might be useful to the reader to clarify some fundamental issues about this topic before

the presentation of the adopted algorithm, which represents the method of resolution of all the problems formulated in this thesis. Furthermore, the presentation of other existing methods allows the reader to better understand advantages, drawbacks and problematics which may arise in this interesting and complex field of research.

Chapter 3: CAO Algorithm

The detailed description of the algorithm adopted and analyzed in this thesis, the Cognitive-based Adaptive Optimization (CAO) algorithm, is presented in this chapter. In particular, several aspects that in our opinion make this method very useful in many practical applications in mobile and cooperative robotics are enumerate and discussed. After the algorithm strategy description, a mathematically rigorous proof of convergence is provided. Firstly is discussed the unconstrained case and the theorem of convergence, as presented in [63], is reported. Then, this result has been extended for the constrained case and the same converge properties are proven.

Chapter 4: Optimal Surveillance Coverage - 2D

As a first step toward the final implementation on the MAVs swarm, we firstly formulate the surveillance coverage problem and test the CAO algorithm for a 2D both convex and non-convex region [91], [90]. The motivation of this choice is that this is a simpler case and it has been well studied in the literature. This allows us to better interpret the simulation results and identify possible situations of stuck in local minima. Initially, two main coverage criteria are identified and analyzed separately. For one of them, also a completely different approach is proposed. It is based on the classical and well-known centroidal Voronoi solution (see [20]) combined with the artificial potential field method [95]. Then, considering the other coverage criterion, the adopted CAO algorithm is adapted and applied to find a solution. Finally, as the last contribution of this chapter, a distributed version of the CAO algorithm is proposed, allowing us to eliminate the drawbacks of a centralized approach and to include communication constraints [92]. Many simulations results are shown to validate the proposed methods.

Chapter 5: Optimal Surveillance Coverage - 3D

After the first tests on 2D areas, the problem is extended for a more realistic and of practical interest 3D case [93], [94]. With respect to the previous analyzed case, more constraints are present and a new, more complex, cost function is defined with the aim of taking into account

both the coverage criteria identified in Chapter 4. To test this new objective function and to set the parameters of the system, some preliminary simulations have been carried out by using simulated environments. In order to be as general as possible, not only simple 3D regions have been reproduced but also complex non-convex surfaces. Additionally, also some simulations to prove the scalability of the algorithm with respect to the number of robots are shown. The results of this validation process show the capability of the proposed method to provide a solution for the cooperative surveillance coverage for a completely arbitrary 3D environment.

Chapter 6: Experimental Results

This chapter include the experimental results obtained by using a real swarm of MAVs for surveillance coverage missions in both indoor and outdoor complex environments. This part of the work is in collaboration with all the partners of the sFly project and especially with the ETH of Zurich. In the first part several simulations are carried out by using real data provided by a helicopter [23], [24]. These data have been collected in complex, outdoor regions near Zurich. In particular, some incremental scenarios are provided where the coverage is achieved simultaneously with the mapping. Finally, the results of a real implementation in an outdoor area, performed during the final demonstration of the sFly project, are presented. The aim of this experimentation is to accomplish a search and rescue mission by using three MAVs in a GPS-denied complex environment. Once the region of interest is mapped by the helicopters, the method presented in this thesis is used to obtain the optimal coverage positions, which allow the team to localize a victim positioned in the environment.

Chapter 7: Navigating between People

Finally, in order to further validate the adopted optimization method, as a last contribution of this thesis, it has been considered also a completely different problem: safely navigating a robot in an unknown and complex environment where people are moving and interacting [97]. The robot has the objective to move and reach predefined goal locations respecting humans' comfort. This problem has been formulated in terms of an optimization problem and the CAO algorithm employed to find a solution. Since in this case the environment is assumed to be dynamic, with moving people, the algorithm has been also used to generate a sort of prediction on the unknown people movement, in order to improve the results. This last test shows how, within the robotic field, many different problems can be formulated as optimization problems and that in dynamical, uncertain and/or sensor-based scenarios a stochastic optimization algorithm is very suitable to obtain a solution.

Chapter 2

Stochastic Optimization

Every problem treated in this thesis has been approached as an optimization problem. Optimization might be defined as the science of determining the best solutions to certain mathematically defined problems, which are often models of physical reality. In practice, this means maximizing or minimizing a function by choosing input values among an allowed set. It involves the study of optimality criteria for problems, the determination of algorithmic methods of solution, the study of the structure of such methods and computer experimentation with method both with trial conditions and on real life problems. There is an extremely diverse range of practical applications. One approach is stochastic optimization, in which the search for the optimal solution involves randomness in some constructive way. Stochastic optimization plays a significant role in the analysis, design and operation of modern systems. Methods for stochastic optimization provide a means of coping with case where only information affected by noise is available and coping with models or systems that are highly nonlinear, high dimensional, or otherwise inappropriate for classical deterministic methods of optimization. Stochastic optimization algorithms have broad application to problems in statistics, computer science, engineering, and business. By now algorithms that employ some form of stochastic optimization have become widely available and exploited. Specific applications include for example business (making short- and long-term investment decisions in order to increase profit), aerospace engineering, medicine, and traffic engineering (setting the timing for the signals in a traffic network) and, as in this thesis, mobile robotics.

The aim of this chapter is to present the general problem a stochastic optimization algorithm is called to solve and to describe some of the most famous algorithms present in literature. The proposed survey of existing algorithms follows [112] and [111]. For other references that give general reviews of various aspects of stochastic optimization see for example: [27], [29], [37] and references therein.

2.1 Problem Formulation

Let us introduce some important concepts and notation. Suppose $\mathcal{S}$ is the domain of allowable values for a vector x. The fundamental problem of interest is to find the value(s) of a vector $x \in \mathcal{S}$ that optimizes a scalar-valued objective function $J(x)$. Other equivalent names for J, also used in this thesis, are optimization function, cost function, performance measure, measure-of-effectiveness, fitness function, or simply criterion. For simplicity of exposition, this chapter focuses on the problem of minimization. Note that a maximization problem can be trivially converted to a minimization problem by changing the sign of the criterion.

The problem of minimizing an objective function $J = J(x)$ can be formally represented as finding the set:

$$x^* \equiv \underset{x \in \mathcal{S}}{\operatorname{argmin}}\, J(x) = x^* \in \mathcal{S} : J(x^*) \leq J(x) \quad \forall x \in \mathcal{S}\,, \tag{2.1}$$

where x is the p-dimensional vector of parameters that are being adjusted and $\mathcal{S} \subseteq \mathbb{R}^p$. The "argmin" statement in (2.1) should be read as: $\mathcal{S}^*$ is the set of values $x = x^*$ that minimize $J(x)$ subject to x^* satisfying the constraints represented in the set $\mathcal{S}$. The elements $x^* \in \mathcal{S}^* \subseteq \mathcal{S}$ are equivalent solutions in the sense that they yield identical values of the cost function. In some cases (i.e., differentiable J), the minimization problem can be converted to a root-finding problem of finding x such that

$$g(x) = \frac{\partial J(x)}{\partial x} = 0\,. \tag{2.2}$$

Of course, as later discussed, this conversion must be done with care because such a root may not correspond to a global minimum of J.

The solution set x^* may be a unique point, a countable (finite or infinite) collection of points, or a set containing an uncountable number of points. The tree examples below show simple cases illustrating these types of solution sets.

1. x^* contains unique solution: suppose that $J(x) = x^T x$ and $S = \mathbb{R}^p$, the unique value which minimizes J is $x = 0$. Then, x^* is a single point.

2. x^* has countable (finite or infinite) collection of points: let x be a scalar, $J = \sin x$ and $S = [0, 4\pi]$. Then the minimum is at the points $x^* = \{3\pi/2, 7\pi/2\}$, a countable set with a finite number of elements. On the other hand, if $S = \mathbb{R}$, x^* is a countable set with an infinite number of elements.

3. x^* has uncountable number of points: suppose that $J = (x^T x - 1)^2$ and $S = \mathbb{R}^p$, then

J is minimized when $x^T x = 1$, which is the set of points lying on the surface of a p-dimensional sphere with radius 1. If $p \geq 2$, x^* is an uncountable (but bounded) set.

Unlike the simple illustrative examples here presented, the problems treated in this thesis are sufficiently complex so that is impossible to obtain a closed-form analytical solution to (2.1).

For ease of exposition, this chapter generally focuses on continuous optimization problems, although some of the methods may also be used in discrete problems. In the continuous case, it is often assumed that J is a "smooth" (and also several times differentiable) function of x. Continuous problems arise frequently in robotics applications as well as in many other fields, such as model fitting (parameter estimation), adaptive control, neural network training, signal processing, and experimental design. Discrete optimization (or combinatorial optimization) can be considered as a large subject by itself (resource allocation, network routing, policy planning, etc.).

Another fundamental issue which clearly separates in two categories the optimization algorithms is the difference between global and local optimization. In other words, the capability of an optimization algorithm to distinguish between global and local optima. All other factors being equal, it is obvious that one would always know a globally optimal solution to the optimization problem. However, in practice, it may not be feasible to find a global solution and one must be satisfied with obtaining a local solution. For example, J may be shaped such that there is a clearly defined minimum point over a broad region of the domain $\mathcal{S}$, while there is a very narrow spike at a distant point. If the value of this spike is lower than any point in the broad region, the local optimal solution is better than any nearby x, but it is not the best possible x. In this thesis we consider only local optimization methods. This important distinction will be better explained in the follows.

Finally, as clear from the main topic of this chapter, we can separate optimization in deterministic and stochastic algorithms. As previously stated, we focus on stochastic optimization and to be more precise, in the framework of the optimization theory, we can talk of stochastic optimization if:

- there is random noise on the available values of the optimization function J;
- and/or there is a random choice in the search strategy as the algorithm iterates toward a solution.

These two conditions are in contrast with the classical deterministic optimization where it is assumed that the information about the function to optimize is perfect (noise-free) and this information is used to decide, for each iteration step of the algorithm, a deterministic search strategy. In many practical applications, especially in the robotics domain, like for the cases

treated in this thesis, such information is not available and a deterministic method becomes completely inappropriate.

2.1.1 Local Versus Global Optimization

As already mentioned, one of the main distinctions in optimization theory is between local and global optimization. Usually, with a finite amount of resources, it is only possible to ensure that an algorithm will approach a local minimum. It can be seen that it is easy to construct functions that will fool most of the known algorithms, unless the algorithm is given explicit a priori information about the location of the global solution, which is certainly not a case of practical interest. However, since the local minimum may still yield a significantly improved solution, the local minimum may be a completely acceptable solution for the resources available (human time, computer time, sensors capabilities, etc.) to be employed on the optimization.

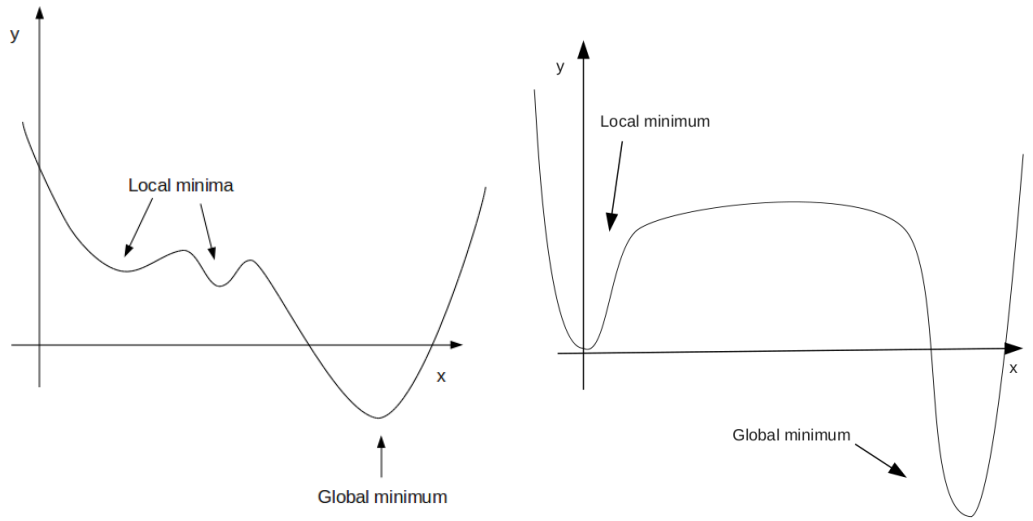
The main drawbacks of local optimization algorithms can be expressed as follows:

- by definition, such algorithms terminate in a local optimum, and there is generally no information about the amount by which this local optimum falls close to a global optimum;
- the obtained local optimum depends on the initial configuration but, at the same time, no guidelines are generally available for its choice.

To avoid some of these drawbacks, a number of possible improvements are possible, even if they are not always feasible in real scenarios:

- execution of the algorithm for a large number of initial configurations, at the cost of an increase of computation time;
- use of information gained from previous runs of the algorithm to improve the choice of an initial configuration for the next run;
- introduction of complex move-generation rules, in order to be able to overcome local optima; this can include the acceptance of moves which correspond to a decrease in the objective function in a limited way, hoping that it will lead to a higher local maxima.

In Fig. 2.1 two examples of local minima are shown. It is possible to see how the dimension and the distance between them is of crucial importance for the possibility of the algorithm to find the global minimum. In Fig. 2.1 (a) three minima are present but the two local minima



(a) (b)

Figure 2.1: Two different simple examples of local minima: in (a) two local minima are present but their dimension is such that the global minimum is easy to find. In (b) there is only one local minimum but a higher barrier, hard to overcome, divides it from the global minimum. In this case the initial position is crucial for the final result.

are of small dimension and very close to the real global optimum, so their identification can be easy regardless the initial configuration. In (b) two minima with a significant different value and with a high barrier dividing them are present. In this case also with a good local optimization algorithm it is very difficult identifying the global minima and the dependence on different starting conditions becomes very strong (for an example see Section 2.3.3).

2.1.2 Some considerations

We enumerate here other key points and limits that can result of critical importance in many practical applications of a stochastic optimization algorithm.

Constraints

The possibility to solve an optimization problem often depends on the presence or not of constraints and a good strategy to deal with them is a critical point for an optimization algorithm. Indeed, constraints on the allowable values of x are always present in every real-life situation. In particular, in a robotic application they can represent, for example, limitations on: environmental constraints (obstacles, limited region of interest, maximum height of flight, etc.), constraints on the robot's movement (maximum velocity, non-holonomic dynamic, etc.),

communication constraints (limited communication capabilities), and so on. A possible classification of different kinds of constraints is between *hard* and *soft*. In hard constraints, no value of x can ever be taken outside of the allowable set. With soft constraints, values of x outside this set are allowed during the search process, but it is required that the final estimate for x lie inside the constrained set. In a typical online implementation the previously listed constraints have to be considered as hard constraints. Generally, the presence of constraints increases significantly the hardness of an optimization problem and ad hoc strategies have to be adopted.

Curse of dimensionality

A fundamental limit for a multivariate ($N > 1$) search is that the volume of search space generally grows geometrically with the dimension. This implies that a naive search in a high-dimensional problem will generally be hopeless. The famous control theorist and mathematician Richard Bellman coined the expression "curse of dimensionality" to describe this phenomenon. In [44], for example, the author gives an illustration in a discrete problem where $N = 10$: if each of the 10 elements of x can take on 10 values, there are 10^{10} possible outcomes. If we randomly sample 10000 values of J uniformly in the domain S, the probability of finding one of the best 500 values for x (which is a much easier problem than finding the unique optimum x^*) is 0,0005. With noisy measurements of J is even worse because it is not known which x value corresponds to the lowest loss value from among the sampled x points. In a robotic framework, this problematic may be translated in a problem of scalability on the number of robots involved in the mission. This issue will be discussed more in details in chapter 5.

Stopping criteria

In search and optimization problems an usual problem to tackle is to develop a good stopping criterion, i.e. a means to decide when the algorithm is close enough to the solution that it can be stopped. Unfortunately, in a general stochastic optimization problem it is almost impossible to find an automatic means of stopping an algorithm with a guaranteed level of accuracy. The fundamental reason is that there will always be a significant region within S that will remain unexplored in any finite number of iterations. Without an a priori knowledge, there is always the possibility that x^* could lie in this unexplored space.

Time-varying problems

In many practical applications, the environment changes over time. Hence, the best solution to a problem in a given time may be very far to the best solution in another instant. Some examples are environments with moving people or vehicles and/or a varying number of robots within the team. In some cases, the algorithms can be explicitly designed to *adapt* to changing conditions and automatically provide a new estimation of the optimal value. In other cases, the only solution is to restart the optimization process and find a new optimum.

Obviously, these are only some of the several problematics that might emerge in a real and complex problem. In the next sections we will give an overview on some of the most common stochastic optimization algorithms. This should allow the reader to better understand the problematics of the stochastic optimization and how such a problem can be approached, starting from the simplest strategy toward more complicate and efficient algorithms. In the following review we assume that the reader is familiar with the most common concepts of deterministic optimization, such as gradient descent algorithms, the role of the Hessian matrix, etc. In any case, we will provide the fundamental references whenever it could be useful.

2.2 Random Search

Probably the simplest methods of stochastic optimization are random search methods and they have been known since at least the 1950s (see [71] for a historical review). Their relative simplicity is an appealing feature to both practitioners and theoreticians but, at the same time, they can be fairly effective in many problems. Indeed, these direct random search methods have a number of advantages relative to most other search methods. The advantages include relative ease of coding in software, the need to only require J measurements (no needs that the gradient of J be computable or even that it exist), reasonable computational efficiency (especially for those direct search algorithms that make use of some local information in their search), broad applicability to non-trivial cost functions and/or to x that may be continuous, discrete, or some hybrid form, and a strong theoretical foundation.

We present here two of the numerous algorithms based on this method: the blind random search and the localized random search.

The first method we discuss is the blind random search. This is the simplest random search method, where the current sampling for x does not take into account the previous samples. That is, this blind search approach does not adapt the current sampling strategy to information that has been stored in the search process. The approach can be implemented in non-recursive form simply by laying down a number of points in $\mathcal{S}$ and taking the value of x corresponding

to the lowest J value as our estimate of the optimum. The approach can be conveniently implemented in recursive form as we illustrate below. The simplest setting for conducting the random sampling of new candidate values of x is when x is a hypercube and we are using uniformly generated values of x. The uniform distribution is continuous or discrete for the elements of x depending on the definitions for these elements. In fact, the blind search form of the algorithm is unique among all general stochastic optimization algorithms in that it is the only one without any adjustable algorithm coefficients that need to be tuned to the problem at hand. The steps for a recursive implementation of blind random search are given below.

2.2.1 Blind Random Search

1. (Initialization) Choose an initial value of x, say $\hat{x}_0 \in \mathcal{S}$, either randomly or deterministically. Calculate $J(\hat{x}_0)$. Set k = 0.

2. Generate a new independent value $x_{new}(k+1) \in \mathcal{S}$, according to the chosen probability distribution. If

$$J(x_{new}(k+1)) < J(\hat{x}_k) \Rightarrow \hat{x}_{k+1} = x_{new}(k+1)\,.$$

   Else, take $\hat{x}_{k+1} = \hat{x}_k$.

3. Stop if the maximum number of J evaluations has been reached or the user is otherwise satisfied with the current estimate for x via appropriate stopping criteria; else, return to Step 1 with the new k set to the former $k+1$.

The above algorithm converges almost surely to x^* under very general conditions (see, e.g., [111], pp. 40-41). Of course, not only the convergence is an indication of the performance of the algorithm. Indeed, for an accurate analysis it is also of interest to examine the rate of convergence. The rate is intended to tell the analyst how close x_k is likely to be to x^* for a given cost of search. While blind random search is a reasonable algorithm when x is low dimensional, it can be shown that the method is generally a very slow algorithm even for state vector x of moderate dimension. This is a direct consequence of the exponential increase in the size of the search space as N increases.

Blind search is the simplest random search in that the sampling generating the new x value does not take account of where the previous estimates of x have been. The random search algorithm below is slightly more sophisticated because the random sampling is a function of the position of the current best estimate for the optimization vector x. In this way, the search

is more localized in the neighborhood of that estimate, allowing for a better exploitation of information that has previously been obtained about the shape of the optimization function.

The localized random search algorithm is presented below. This algorithm was described by Matyas in [78]. Note that the use of the term "localized" here pertains to the sampling strategy and does not imply that the algorithm is only useful for local (versus global) optimization in the sense described previously. In fact, the algorithm has also global convergence properties.

2.2.2 Localized Random Search

1. (Initialization) Pick an initial candidate $\hat{x}_0 \in \mathcal{S}$, either randomly or with prior information. Set $k = 0$.

2. Generate an independent random vector $d_k \in Rp$ and add it to the current x value, $\hat{x}_k$. Check if $\hat{x}_k + d_k \in \mathcal{S}$. If $\hat{x}_k + d_k \notin \mathcal{S}$, generate a new d_k and repeat or, alternatively, move $\hat{x}_k + d_k$ to the nearest valid point. Let $\hat{x}_{new}(k+1)$ equal $\hat{x}_k + d_k \in \mathcal{S}$ or the aforementioned nearest valid point in $\mathcal{S}$.

3. If
$$J(x_{new}(k+1)) < J(\hat{x}_k) \Rightarrow \hat{x}_{k+1} = x_{new}(k+1)\,.$$
Else, set $\hat{x}_{k+1} = \hat{x}_k$.

4. Stop if the maximum number of J evaluations has been reached or the user is otherwise satisfied with the current estimate for x via appropriate stopping criteria; else, return to Step 1 with the new k set to the former $k + 1$.

For continuous problems, [78] and others have used the (multivariate) normal distribution for generating d_k. However, the user is free to set the distribution of the deviation vector d_k. The distribution should have mean zero and each component should have a variation (e.g., standard deviation) consistent with the magnitudes of the corresponding x elements. This allows the algorithm to assign roughly equal weight to each of the components of x as it moves through the search space. Although not formally allowed in the convergence theory, it is often advantageous in practice if the variability in d_k is reduced as k increases. This could be a fundamental point for the success of the optimization process because it allows one to focus the search more tightly in the surrounding of the estimated location of the solution (as expressed by the location of our current estimate $\hat{x}_k$).

2.3 Stochastic Approximation

Stochastic approximation (SA) is a very important topic in the framework of stochastic optimization. Robbins and Monro introduced in [99] SA as a general root-finding method when only noisy measurements of the underlying function are available. Let us now discuss some aspects of SA as applied to the more specific problem of root-finding in the context of optimization. With a differentiable cost function $J(x)$, recall the familiar set of N equations and N unknowns for use in finding a minimum x^*:

$$g(x) = \frac{\partial J(x)}{\partial x} = 0\,. \tag{2.3}$$

Of course, side conditions are required to guarantee that a root of (2.3) is a minimum, not a maximum or saddle point. Note that (2.3) is nominally only directed at local optimization problems. There are a number of approaches for solving the problem represented by (2.3) when direct and usually noisy measurements of the gradient g are available. These typically go by the name of stochastic gradient methods (e.g., [111], Chap. 5). In contrast to the stochastic gradient approach but consistent with the emphasis in the random search let us focus on SA when only measurements of J are available. However, we consider noisy measurements of J. To motivate the general Stochastic Approximation approach, first recall the familiar form for the unconstrained deterministic steepest descent algorithm for solving (2.3):

$$\hat{x}_{k+1} = x_k - a_k g(\hat{x}_k)\,, \tag{2.4}$$

where the gain (or step size) satisfies $a_k > 0$ (see, e.g., [7]). This algorithm requires exact knowledge of g. Steepest descent will converge to x^* under certain fairly general conditions. A notable variation of steepest descent is the Newton-Raphson algorithm (sometimes called Newton's method; e.g., [7]), which has the form

$$\hat{x}_{k+1} = x_k - a_k H(\hat{x}_k)^{-1} g(\hat{x}_k)\,, \tag{2.5}$$

where $H(\cdot)$ is the Hessian (second derivative) matrix of J. Under more restrictive conditions, the Newton-Raphson algorithm has a much faster rate of convergence to x^* than steepest descent. However, with its requirement for a Hessian matrix, it is generally more challenging to implement.

Unlike the case of steepest descent, it is assumed here that we have no direct knowledge

of the gradient g. The recursive procedure of interest is in the general SA form:

$$\hat{x}_{k+1} = x_k - a_k \hat{g}(\hat{x}_k)\,, \tag{2.6}$$

where $\hat{g}(\hat{x}_k)$ is the estimate of g based on measurements of the cost function. Under appropriate conditions, the iteration in (2.6) converges to x^* in some stochastic sense (usually almost surely).

In the next sections we discuss two popular SA methods for carrying out the optimization task using noisy measurements of the objective function. The first one is the traditional finite-difference SA method (FDSA); then, we describe more in detail the well known simultaneous perturbation SA method (SPSA).

2.3.1 Finite Difference Stochastic Approximation

The more relevant part of (2.6) is the gradient approximation $\hat{g}_k(\hat{x}_k)$. One of the classical methods to form the approximation is the finite-difference method. Expression (2.6) with this approximation represents the finite-difference SA (FDSA) algorithm. One-sided gradient approximations involve measurements $J(\hat{x}_k)$ and $J(\hat{x}_k + \text{perturbation})$, while two-sided approximations involve measurements of the form $J(\hat{x}_k \pm \text{perturbation})$. The two-sided FD approximation for use with (2.6) has the following form:

$$\hat{g}_k(\hat{x}_k) = \begin{bmatrix} \frac{J(\hat{x}_k + c_k\epsilon_1) - J(\hat{x}_k - c_k\epsilon_1)}{2c_k} \\ \vdots \\ \frac{J(\hat{x}_k + c_k\epsilon_N) - J(\hat{x}_k - c_k\epsilon_N)}{2c_k} \end{bmatrix} \tag{2.7}$$

where ϵ_i denotes a vector with a 1 in the ith place and 0's elsewhere and $c_k > 0$ defines the differences magnitude. The pair $\{a_k, c_k\}$ are the gains sequences for the FDSA algorithm. The two-sided form in (2.7) is the obvious multivariate extension of the scalar two-sided form in [55]. The initial multivariate method in [13] used a one-sided approximation.

It is of fundamental importance to determine conditions such that $\hat{x}_k$ as shown in (2.7) converge to x^* in some appropriate stochastic sense. The convergence theory for the FDSA algorithm is similar to standard convergence theory for the root-finding SA algorithm of Robbins and Monro ([99]). Additional difficulties, however, arise due to a bias in $\hat{g}(\hat{x}_k)$ as an estimator of $g(\hat{x}_k)$. That is, standard conditions for convergence of SA require unbiased estimates of g at all k. On the other hand, $\hat{g}(\hat{x}_k)$ as shown in (2.7), is a biased estimator, with the bias having a magnitude of order c_k^2. We will not present here the details of the convergence theory (see for example [66], [111]). However, let us note that the standard conditions on the

gain sequences are:

$$a_k > 0,\ c_k > 0,\ a_k \to 0,\ c_k \to 0,\ \sum_{k=0}^{\infty} a_k = \infty,\ \sum_{k=0}^{\infty} \frac{a_k^2}{c_k^2} < \infty\,. \tag{2.8}$$

The choice of this gain sequence is critical to the performance of the method. Commons forms are:

$$a_k = \frac{a}{(k+1+A)^{\alpha}}, \quad c_k = \frac{c}{(k+1)^{\gamma}}\,, \tag{2.9}$$

where the coefficients a, c, α and γ are strictly positive and $A \geq 0$. The user must choose these coefficients, a process usually based on a combination of the theoretical restrictions above, trial-and-error numerical experimentation, and basic problem knowledge.

2.3.2 SPSA Algorithm

The FDSA algorithm is a standard SA method for carrying out optimization with noisy measurement of the objective function. However, as the dimension N grows large, the number of optimization measurements required may become prohibitive. That is, each two-sided gradient approximation requires $2N$ measurements. The simultaneous perturbation SA (SPSA) method [108] requires only two measurements per iteration to form a gradient approximation independent of the dimension N. This point, very important also later in this thesis, deserves a further discussion. The relevance of reducing the number of measurements can be crucial not only when the state dimension is huge, but also in other cases when the energy consumption is a strong constraint to take into account. And this is a very common situation in mobile robotics and in particular using micro aerial vehicles. The reason is that an extra measurement means, when the objective function is known only by sensors, an extra unnecessary movement for the system.

For the two-sided SP gradient approximation, this leads to:

$$\begin{aligned} \hat{g}_k(\hat{x}_k) &= \begin{bmatrix} \frac{J(\hat{x}_k + c_k\Delta_k) - J(\hat{x}_k - c_k\Delta_k)}{2c_k\Delta_{k1}} \\ \vdots \\ \frac{J(\hat{x}_k + c_k\Delta_k) - J(\hat{x}_k - c_k\Delta_k)}{2c_k\Delta_{kN}} \end{bmatrix} \\ &= \frac{J(\hat{x}_k + c_k\Delta_k) - J(\hat{x}_k - c_k\Delta_k)}{2c_k} \left[\Delta_{k1}^{-1}, \Delta_{k2}^{-1}, \ldots, \Delta_{kN}^{-1}\right]^T \end{aligned} \tag{2.10}$$

where the zero-mean $N-$dimensional random perturbation vector:

$$\Delta_k = [\Delta_{k1}^{-1}, \Delta_{k2}^{-1}, \ldots, \Delta_{kN}^{-1}]^T,$$

has a user-specified distribution satisfying certain conditions and c_k is a positive scalar (as with the FDSA). Because the numerator is the same in all N components of $\hat{g}_k(\hat{x}_k)$, the number of function measurements needed to estimate the gradient in SPSA is two, regardless of the dimension N.

The choice of the distribution for generating the Δ_k is important to the performance of the algorithm. The standard conditions for the elements Δ_{ki} are that the $\{\Delta_{ki}\}$ are independent for all k, i, identically distributed for all i at each k, symmetrically distributed about zero and uniformly bounded in magnitude for all k. In addition, there is an important inverse moments condition:

$$E\left(\left|\frac{1}{\Delta_{ki}}\right|^{2+2\tau}\right) \leq C$$

for some $\tau > 0$ and $C > 0$. The role of this condition is to control the variation of the elements of $\hat{g}_k(\hat{x}_k)$. One simple and popular distribution that satisfies the inverse moments condition is the symmetric Bernoulli ± 1 distribution.

It has been proved that, under reasonably general conditions (see [108],[111]), the SPSA and FDSA algorithms achieve the same level of statistical accuracy for a given number of iterations even though SPSA uses only $1/N$ times the number of function evaluations of FDSA (since each gradient approximation uses only $1/N$ the number of function evaluations). An intuitive explanation of this remarkable performance advantage for the SPSA is that the algorithm provides a sort of average (controlled by the a_k term) of the gradient approximation across iterations. Since the SPSA gradient approximation is an almost unbiased estimator of the gradient [110], the errors in the approximation tend to average out over the long run of iterations. This $1/N$ advantage for SPSA can result of fundamental importance in savings for a complex (large N) application when the objective function measurements are costly to obtain (See Fig. 2.2).

An accelerated form of SPSA is reported in [109]. This approach extends the SPSA algorithm to include second-order (Hessian) effects with the aim of accelerating convergence in a stochastic analogue to the deterministic Newton-Raphson algorithm. Like the standard (first-order) SPSA algorithm, this second-order algorithm is simple to implement and requires only a small number - independent of N - of cost function measurements per iteration (no gradient measurements, as in standard SPSA). In particular, only four measurements are required to estimate the objective function gradient and inverse Hessian at each iteration (and one additional measurement is sometimes recommended as a check on algorithm behavior). The algorithm is implemented with two simple parallel recursions: one for x and one for the Hessian matrix of $J(x)$. The recursion for x is a stochastic analogue of the well known Newton-Raphson algorithm for deterministic optimization. The recursion for the Hessian matrix is simply a

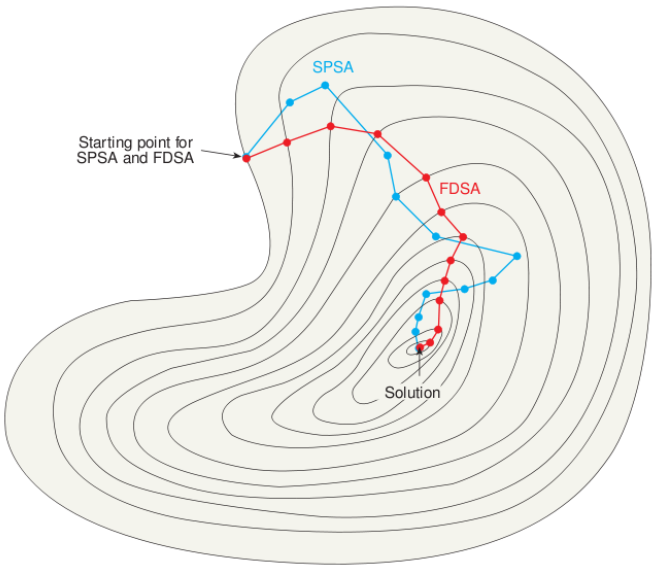


Figure 2.2: Example of search path for SPSA and FDSA in a $N = 2$ problem ([110]).

recursive calculation of the sample mean of per-iteration Hessian estimates formed using SP-types ideas.

In conclusion, the SPSA algorithm has proven to be an efficient optimization tool in many complex applications. In addition, a large number of works on the theory and practice of SPSA has accumulated, generalizing the convergence form of the algorithm and proving their properties, giving instruction on how to set up and run the algorithm and describing numerous successful applications.

2.3.3 Global optimization using SPSA

As we already stated, this chapter focuses only on local optimization and we do not address our attention to global optimization algorithms which often are very costly, or even impossible, to be implemented in real applications. However, we show here how is possible to modify a SA algorithm to cope with local minima problems and why this kind of approaches can outperform a deterministic method in these cases.

In several works (e.g. see [64], [35], [113] and [122]) has been proposed a strategy to overcome possible local minima injecting random noise in the right-hand side of the basic SA updating step in (2.6), or also in the deterministic version (2.4). For example, in the latter case, it has the following form:

$$\hat{x}_{k+1} = \hat{x}_k - a_k g(\hat{x}_k) + d_k \omega_k \,, \tag{2.11}$$

where d_k are appropriately selected and ω_k are random (usually Gaussian) p-dimensional vector satisfying certain conditions. This strategy is based on the intuitive idea that injecting such a random noise might allow the algorithm to escape x neighborhoods that produce local minima of $J(x)$, especially in the first iterations of the algorithms. It has been proven that with some conditions on this noise, the algorithm converge, in some sense, to the global minimum [64], [26]. On the other hand, a similar approach requires more tuning of the extra terms d_k and ω_k, and retards the convergence in proximity of the solution.

The use of SPSA for global minimization among multiple local minima is discussed by Maryak and Chin in [76], [77]. Firstly, they propose this common way of converting SA algorithms to global optimizers through the additional "bounce" introduced into the algorithm. Then, the authors also show that basic SPSA without injected noise may, under certain conditions, be a global optimizer. Formal justification for this result follows because the random error in the SP gradient approximation operates in a way that is statistically equivalent to the injected noise mentioned above. In other words we can express the standard SPSA recursion as follows

$$\hat{x}_{k+1} = \hat{x}_k - a_k(g(\hat{x}_k) + \omega_k^*)\,, \tag{2.12}$$

where $\omega_k^* \equiv \hat{g}_k(\hat{x}_k) - g(\hat{x}_k)$ is the effective noise. Even if some important properties are not verified in this case (ω_k^* is not a vector of independent, identically distributed standard Gaussian noise), under certain conditions, the basic SPSA algorithm can achieve global convergence without injecting noise.

2.3.4 Constrained SPSA

The SPSA algorithm as here presented does not include constraints on the optimization variables. As previously stated, the presence of constraints can complicate significantly an optimization problem, but on the other hand dealing with unconstrained problems is almost impossible in usual practical applications. We illustrate now the constrained version of the SPSA algorithm, as presented by Sadegh in [102], which can deal with inequality constraints. Hence, the problem it is called to solve is:

$$\min_{x \in S} J(x) \tag{2.13}$$

where S is the feasible space and $J(x)$ is continuously differentiable on an open set containing S. We assume that the set

$$G = \{x : q_i(x) \leq 0\,, i = 1 \ldots s\}$$

is non-empty and bounded, and the constraint functions $q_i(x)$ are continuously differentiable $\forall i$. At each $x \in \partial S$, where ∂ denotes the boundary, the gradients of the active constraints are linearly independent. Furthermore, there exists an $\epsilon < 0$ such that the set

$$G = \{x : q_i(x) \leq r\,, i = 1 \ldots s\} \tag{2.14}$$

is non-empty for $\epsilon \leq r < 0$, i.e. the set S has a non-empty interior.

Let $\hat{x}_k$ denote the estimate for x at the k-th iteration, and for all $x \in \mathbb{R}^p$ let $\Upsilon(x)$ be the nearest point to x on S, where the norm is defined as the usual Euclidean norm. So the projection algorithm, extending the expression in (2.6), has the general form:

$$\hat{x}_{k+1} = \Upsilon(\hat{x}_k - a_k \hat{g}(\hat{x}_k))\,. \tag{2.15}$$

In the unconstrained problem, $\hat{g}$ has the standard form g^{SP} as in (2.10). Such an approximation cannot be directly adopted here, since it may happen that $x_k \in S$ but $\hat{x}_k \pm c_k\Delta_k \notin S$. Especially, in the case $x_k \in \partial S$, there is always a random direction Δ_k such that $\hat{x}_k \pm c_k\Delta_k \notin S$, no matter how small is the selected gain c_k. Note that the case $x_k \in \partial S$ is expected to occur frequently in the relevant situation where the true optimum belongs to the boundary of the feasible domain. Except for simulation-based optimization cases, function evaluations might involve real measurements and it is usually not allowed taking measurements outside the feasible space. To overcome this problem, we further project $\hat{x}_k$ onto a closed set S_k contained within S to obtain $\Upsilon_k(x_k)$, which will be used to compute an SP gradient approximation at the k-th iteration. If the distance d_k between the nearest points on ∂S and ∂S_k is equal to or larger than $c_k\alpha_0$, where α_0 is a bound of the perturbation Δ_k, then $\Upsilon(\hat{x}_k) \pm c_k\Delta_k \in S$, ensuring that the SP approximation to the gradient at $\Upsilon(\hat{x}_k)$ requires no function measurement outside S. The SP gradient approximation at $\Upsilon(\hat{x}_k)$ obviously introduces an extra error term relative to the SP gradient approximation at x_k. However, if $S_k \to S$ as $k \to \infty$ then continuous differentiability of $J(x)$ yields that the extra error term tends to zero.

For this algorithm, the following convergence proprieties hold. Let the previous assumptions and the conditions of Lemma 1 of [108] hold where all regularity conditions on J hold on an open set containing S. Then, under the projection algorithm (2.15), where $\hat{g}(\hat{x}_k) = g_k^{SP}(\Upsilon(\hat{x}_k))$, as $k \to \infty$

$$\hat{x}_k \to KT \text{ a.s.}, \tag{2.16}$$

where KT is the set of Kuhn-Tucker points, i.e. the set of points x where there are $\lambda_i \geq 0$

such that

$$g(x) + \sum_{i:q_i(x)=0} \lambda_i \frac{dq_i(x)}{dx} = 0\,.$$

Proof. Decompose the error

$$\hat{g}_k(\hat{x}_k) - g(\hat{x}_k) = g_k^{SP}(\Upsilon_k(\hat{x}_k)) - g(\hat{x}_k)$$

into a sum of

$$\begin{aligned} b_k^1 &= E(g_k^{SP}(\Upsilon_k(\hat{x}_k))|\hat{x}_k) - g(\Upsilon_k(\hat{x}_k))\,, \\ e_k &= g_k^{SP}(\Upsilon_k(\hat{x}_k)) - E(g_k^{SP}(\Upsilon_k(\hat{x}_k))|\hat{x}_k)\,, \\ b_k^{11} &= g(\Upsilon_k(\hat{x}_k)) - g(\hat{x}_k)\,. \end{aligned} \tag{2.17}$$

Identically to the proof of Lemma 1 and Proposition 1 in [108], it can shown that:

1. $\sup_k |b_k^1|, \infty$ and $b_k^1 \to 0$ almost surely as $k \to 0$;
2. $\lim_{k\to 0} Pr\left(\sup_{m\geq k} |\sum_{i=k}^m a_i e_i| \geq \eta\right)$ for any $\eta > 0$, where $Pr(\cdot)$ denotes probability; moreover, since S is bounded, $S_k \to S$, and $J(x)$ is continuously differentiable at all $x \in S$;
3. $\sup_k |b_k^{11}| < \infty$ and $b_k^{11} \to 0$ as $k \to \infty$.

Then, the assumption of Theorem 5.3.1 of Kushner and Clark [65] are satisfied and the result follows. This result is of fundamental importance also for our intents. Indeed, as explained in the next section, we adopt a similar strategy to make our algorithm able to deal with constrained problems.

We finally mention that the same assumptions we have made also hold for the basic (two-sided) FDSA algorithm. Adjusting the S_k to the component-wise perturbation of the parameters for gradient approximations, it then follows that the same convergence proof holds for the projection FDSA.

2.4 Conclusions

In this chapter we have introduced the important concept of stochastic optimization. A mathematical formulation of the problem has been provided and many fundamental aspects, problematics and limitations of these kinds of optimization problems have been discussed. Then, a short review of the most common algorithms has been presented, to allow the reader to

better understand the possible existing strategies to find a solution. A further aim of this chapter was to give more emphasis on the characteristics which mainly differentiate algorithms, especially for real implementations: concepts like scalability, adaptivity, number of measurements (or robots' displacements) required are of fundamental importance in mobile cooperative robotics. A particular attention has been given to the SPSA algorithm, an approach very close to the algorithm we propose, describing its proprieties and advantages. In the next chapter we will present in detail the stochastic optimization algorithm we adopted in this thesis, the motivations of this choice and its main properties of convergence.

Chapter 3

CAO Algorithm

In this chapter we describe in detail the algorithm we use to obtain the main results of this thesis: the cognitive-based, adaptive optimization algorithm (CAO). Then, we also show how this approach can be appropriately adapted and extended so that it is applicable to the problem of multi-robot coverage treated in this thesis. The CAO methodology, which was recently introduced by Kosmatopoulos in [63], [60], is a stochastic optimization algorithm which possesses the capability of being able to efficiently handle problems for which an analytical expression of the function to optimize is not available, but the numerical values of this function are available at each iteration of the algorithm employed to optimize it. As a result, it perfectly suits for multi-robot optimal coverage in non-convex environments, where the analytical form of the function to be optimized is unknown but the function is available for measurement (through the robots' sensors) for each multi-robot configuration. In the last section we will also discuss the convergence properties of the algorithm, providing rigorous proofs.

We start quickly defining the optimization problem associated with the practical mission we want to accomplish to motivate our choice of using this algorithm. Let us consider the problem where M robots are involved in a coverage task, attempting to optimize a given coverage criterion. The coverage criterion can be expressed like a function of the robots' positions or poses (positions and orientations), i.e.,

$$J_k = \mathcal{J}\left(x_k^{(1)}, \ldots, x_k^{(M)}\right) \tag{3.1}$$

where $k = 0, 1, 2, \ldots$ denotes the time-index, J_k denotes the value of the coverage criterion at the k-th time-step, $x_k^{(1)}, \ldots, x_k^{(M)}$ denote the position/pose vectors of robots $1, \ldots, M$, respectively, and $\mathcal{J}$ is a nonlinear function which depends, apart from the robots' positions/poses, on the particular environment where the robots live; for instance, in the 2D case the function $\mathcal{J}$ depends on the location of the various obstacles that are present, while in the more com-

plex 3D case with flying robots monitoring a terrain, the function $\mathcal{J}$ depends on the particular terrain morphology.

Due to the dependence of the function $\mathcal{J}$ on the particular environment characteristics, the explicit expression of the function $\mathcal{J}$ is not known in most 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 thesis, the current value of the coverage criterion can be estimated from the robots' sensor measurements. In other words, at each time-step k, an estimate of J_k is available through robots' sensor measurements,

$$J_k^n = \mathcal{J}\left(x_k^{(1)}, \ldots, x_k^{(M)}\right) + \xi_k \tag{3.2}$$

where J_k^n denotes the estimate of J_k and ξ_k denotes the noise introduced in the estimation of J_k due to the presence of noise in the robots' sensors. Note that, even though 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' sensor noise is AWNG: as $\mathcal{J}$ is a nonlinear function of the robots' positions/poses (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 coverage algorithms have additionally to deal with the problem of restricting the robots' positions so that obstacle avoidance as well as robot formation constraints are met. In other words, at each time-instant k, the vectors $x_k^{(i)}, i = 1, \ldots, M$ should satisfy a set of constraints which, in general, can be represented as follows:

$$\mathcal{C}\left(x_k^{(1)}, \ldots, x_k^{(M)}\right) \leq 0 \tag{3.3}$$

where $\mathcal{C}$ is a set of nonlinear functions of the robots' positions/poses. 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 coverage algorithm is provided with information whether a particular selection of robots' positions/poses satisfies or violates the set of constraints (3.3).

Given the mathematical description presented above, the multi-robot coverage problem can be mathematically described as the problem of moving $x_k^{(1)}, \ldots, x_k^{(M)}$ to a set of positions/poses that solves the following constrained optimization problem:

$$\begin{aligned} &\text{minimize} \quad (3.1) \\ &\text{subject to} \quad (3.3)\,. \end{aligned} \tag{3.4}$$

As already noticed, the difficulty in solving, in real-time and in real-life situations, the constrained optimization problem (3.4) lies in the fact that explicit expressions for the functions $\mathcal{J}$ and $\mathcal{C}$ are not available. To circumvent this difficulty, the CAO approach, appropriately modified to be applicable to the problem in hand, is adopted. Indeed this algorithm is capable of efficiently dealing with optimization problems for which the explicit forms of the objective function and constraints are not known, but noisy measurements/estimates of these functions are available at each time-step. In the following, we describe the CAO approach as applied to a completely general multi-robot problem which involves the optimization of an objective function.

It has to be emphasized that the CAO algorithm presented here is an extension of the CAO versions presented and analyzed in [63, 60]. The main difference is that, while in these works the authors address the unconstrained version of the problem (3.4), in the present thesis the CAO approach of [63, 60] has to be extended so that it efficiently takes care of the constraints (3.3). In order to do so, the CAO approach of [63, 60] has been modified by a special, but yet simple, projection mechanism. Theorem 1 establishes that the introduction of such a projection mechanism does not destroy the nice properties of the unconstrained version of the CAO approach; as a matter of fact, according to Theorem 1 presented below, the CAO algorithm used in this thesis is proven to be approximately a projected gradient-descent algorithm, while the ones of [63, 60] have been established to be approximate unconstrained gradient-descent algorithms.

We finally mention that the CAO approach extends the popular Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm, extensively described in the previous chapter. The difference between the SPSA and the CAO approach is that SPSA employs an approximation of the gradient of an appropriate objective function using only the most recent available data, while the CAO approach employs linear-in-the-parameters approximators that incorporate information of a user specified time window of the past experiments together with the concept of candidate perturbations for efficiently optimizing the unknown function. Comparative evaluations that were performed on complicated optimization problems have shown that CAO exhibits significantly better convergence properties than SPSA [61, 63, 60]. Moreover, CAO was shown to exhibit satisfactory (local) convergence characteristics in particular problems where SPSA failed to provide convergent solutions for any choice of its design parameters, [61, 60].

It is important to note that both the CAO and the SPSA do not create an approximation or estimation of the environmental characteristics, like for example the obstacles location and geometry; instead, they on-line produce a only local approximation of the unknown cost function the robots are called to optimize. For this reason, they require simple, and scalable,

approximation schemes to be employed.

3.1 Preliminaries

3.1.1 Notation

A function f is said to be C^m, where m is a positive integer, if it is uniformly continuous and its first m derivatives are uniformly continuous. The notation $\mathrm{vec}\,(A, B, \dots,)$, where $A, B, \dots$ are scalars, vectors or matrices, is used to denote a vector whose elements are the entries of $A, B, C, \dots$ (taken column-wise). $Z_+, \Re_+$ denote the set of nonnegative integers and nonnegative real numbers, respectively. For a vector $x \in \Re^n$, $|x|$ denotes the Euclidean norm of x (i.e. $|x| = \sqrt{x^\tau x}$), while for a matrix $A \in \Re^{n^2}$, $|A|$ denotes the induced matrix norm of A. $\nabla J(x, \theta)$ denotes the gradient of $J(x, \theta)$ with respect to x. Also, if $x_k, k \in Z_+$ is a vector sequence, then $x_{\{\ell,\dots,k\}}$ for $\ell < k$ is used to denote the vectors $x_\ell, x_{\ell+1}, \dots, x_k$. The notation $O(\cdot)$ is used to denote the standard "order of" notation. Finally, we will say that, see [28], a function $\chi : \Re_+ \mapsto \Re_+$ is of class K_∞ (symbolically, $\chi \in \mathsf{K}_\infty$) when χ is continuous, strictly increasing, $\chi(0) = 0$ and $\chi(r) \to \infty$ as $r \to \infty$.

3.1.2 PℓUAs

In this chapter, we make use of a special family of function approximators, the Polynomial-like Universal Approximators (PℓUAs) for the approximation/estimation of unknown functions. For this reason, some preliminaries are needed regarding PℓUAs and their approximation capabilities. More precisely, let F: $\mathbb{R}^{n_1} \to \mathbb{R}^{n_2}$ be an unknown function to be approximated over a compact set $\Xi \subset \mathbb{R}^{n_1}$ and let $\|F\|_\Xi$ denote one of the following norms

$$\|F\|_\Xi = \sqrt{\int_\Xi |F|^2 \, dx} \quad \text{or} \quad \|F\|_\Xi = \sup_{x \in \Xi} |F(x)| \, .$$

A PℓUA used for the approximation of F takes the form:

$$\hat{F}(x) = \vartheta^\tau \phi(x) = \vartheta_1^\tau \phi_1(x) + \vartheta_2^\tau \phi_2(x) + \ldots + \vartheta_L^\tau \phi_L(x) \tag{3.5}$$

where $\hat{F}$ denotes the approximation of F, ϑ denotes the matrix of parameter estimates, L denotes the size of the PℓUA estimator (3.5) and, finally, ϕ is a non-linear smooth vector function of PℓUA's regressor terms; the entries of the regressor vector ϕ are multi-linear functions of the entries $S(x_i)$, where S may be any smooth monotone function. In other words, the entries

$\phi_i(x)$ of the regressor vector are of the form

$$\phi_i(x) = S(x_i)^{d_{i,1}} S(x_2)^{d_{i,2}} \dots S(x_{n_1})^{d_{i,n_1}} \tag{3.6}$$

where $d_{i,j}$ are nonnegative integers.

Let us fix a PℓUA of the form (3.5) with L regressor terms of the form (3.6) and the compact subset Ξ. Then, the optimal parameter matrix θ^* and the optimal modeling error ν with respect to L, ϕ, F and Ξ are defined as follows:

$$\vartheta^* \stackrel{\triangle}{=} W(L, \phi, F, \Xi) \stackrel{\triangle}{=} \underset{\vartheta}{\operatorname{argmin}} \|F(x) - \vartheta^\tau \phi(x)\|_\Xi \tag{3.7}$$

and

$$\nu \stackrel{\triangle}{=} \mathcal{N}(L, \phi, F, \Xi) \stackrel{\triangle}{=} F(x) - \vartheta^{*\tau} \phi(x)\,. \tag{3.8}$$

Standard results in theory of approximation using polynomial-like function (see e.g. [75] and the references therein), can be used to establish that the following property holds:

- Consider a PℓUA of the form (3.5) with L regressor terms of the form (3.6). Then, there exists a non-decreasing scalar function $\eta : \mathbb{R} \to \mathbb{R}_+$ with $\eta(0) = 0$ such that:

$$\|\nu\|_\Xi \leq \eta\left(\frac{1}{L}\right)\,. \tag{3.9}$$

For convenience of notation, we will use the notation $\nu = \Omega\left(\frac{1}{L}\right)$ to indicate that there exists a bound of the form (3.9) that relates the magnitude of the modeling error term with the number of regressor terms used. In other words: to indicate that the PℓUA modeling error term ν can become arbitrarily small by increasing the size of the PℓUA.

3.2 The Proposed Algorithm

To formally describe the proposed algorithm, let us consider a general optimization problem expressed by the objective function J which is known by means of measurements:

$$J_k \equiv J(x_k, \theta_k)\,, \tag{3.10}$$

where x_k denotes the value of the control parameters vector at the k-th algorithm iteration and the vector θ_k may correspond to signals that are not available for measurements (e.g. sensor noise, un-measurable disturbances, etc.) as well as signals that are available for measurements

but they are not controllable (e.g. system states, measurable disturbance, references signals, etc.).

Before to discuss the algorithm in details, we firstly present a formal definition of the two basic criteria that will be used in this thesis for the evaluation of Cognitive-based Adaptive Optimization algorithms.

Consider the unknown function J defined in (3.10) and consider an iterative algorithm which, at the k-th iteration, calculates the current vector x_k as a function of (a) the past values of the parameter vectors $x_0, x_1, \ldots, x_{k-1}$, (b) the past values of the objective function measurements $J_0, J_1, \ldots, J_{k-1}$, (c) some estimates/measurements $\bar{\theta}_0, \bar{\theta}_1, \ldots, \bar{\theta}_{k-1}$ of the past values of the exogenous vectors $\theta_0, \theta_1, \ldots, \theta_{k-1}$, respectively and (d) a prediction $\bar{\theta}_k$ of the current exogenous vector θ_k. We will say that the aforementioned algorithm is a (σ, τ_k)-*Adaptive Optimization algorithm*, where σ is a nonnegative scalar and τ_k is a nonnegative scalar sequence, if

1. $\limsup_{k\mapsto\infty} |\nabla J(x_k, \theta_k)| \leq \sigma$ if θ_k is a bounded deterministic vector sequence or $\limsup_{k\mapsto\infty} E[|\nabla J(x_k, \theta_k)| \; |G_{k-1}] \leq \sigma$ if θ_k is a stochastic vector sequence, where G_{k-1} is an appropriately defined σ-field (see e.g. Theorem 3 in sec. 3.3), and

2. $J_k < J_{k-1} + \tau_k, \; \forall k \in Z_+$.

The criterion (1) is used to evaluate the steady-state characteristics of adaptive optimization algorithms (how close to a local minimum of J the algorithm converges), while criterion (2) is used for the evaluation of the worst-case transient performance characteristics of the adaptive optimization algorithms. Apparently, one wishes to make σ and τ_k equal to zero; however, since J is an unknown function and, moreover, in most applications the vectors θ_k are not exactly known (or even worse: in some applications these terms are totally unknown) it is impossible, in general, to make σ and τ_k equal to zero. In most applications, the best that an AO algorithm can do is to guarantee that the magnitude of the terms σ and τ_k is upper bounded by some bounds that depend on θ_k and the estimation/prediction accuracy $\theta_k - \bar{\theta}_k$.

Remark 1 [Objective function J**]** Note that criteria (1) and (2) are used in order to evaluate the performance of an adaptive optimization algorithm and do not impose any assumption on the unknown objective function J. $\diamond$

Two further remarks are in order, before we proceed to the presentation of the proposed algorithm.

Remark 2 [Availability of $\bar{\theta}_k$**]** We assumed that some estimates/predictions $\bar{\theta}_0, \ldots, \bar{\theta}_k$ of the exogenous vectors $\theta_0, \ldots, \theta_k$ are available. If such estimates/predictions are not available, as it happens in many practical applications, all the results of this thesis are still valid by setting $\bar{\theta}_k = 0, \forall k$ and $c_{\bar{\theta}} = c_\theta$, where $c_{\bar{\theta}}, c_\theta$ are defined in Theorem 3. $\diamond$

Remark 3 [τ_k for Standard SA Algorithms] In order to evaluate the performance of standard SA algorithms under criterion (2), let us consider a case where $|\nabla J(x_{k-1}, \theta_k)| = B_k$; then, it can be seen that in the case of standard SA algorithms, such as the SPSA or the Random Directions Kiefer-Wolfowitz [65] algorithms, the term τ_k is given by:

$$\tau_k = \alpha_k B_k + b_1 \alpha_k^2 + b_2$$

with b_1, b_2 being two positive terms that depend on θ_k and $\theta_k - \theta_{k-1}$. In other words, in the case where x_{k-1} is quite far from a local minimum of J, standard SA algorithms cannot avoid large "spikes" of J_k. $\diamond$

We are now ready to proceed to the presentation of the proposed algorithm. As a first step, the CAO approach makes use of a linear in the parameters function approximator in order to generate, at each algorithm step k, an estimate of the unknown function $\mathcal{J}$ as follows:

$$\hat{J}_k \left(x_k^{(1)}, \ldots, x_k^{(M)}, \theta \right) = \vartheta_k^\tau \phi \left(x_k^{(1)}, \ldots, x_k^{(M)}, \bar{\theta} \right) . \tag{3.11}$$

Here $\hat{J}_k \left(x_k^{(1)}, \ldots, x_k^{(M)}, \theta \right)$ denotes the approximation/estimation of $\mathcal{J}$ generated at the k-th time-step, $\bar{\theta}$ denotes an estimate of the actual exogenous vector θ, ϕ denotes the nonlinear vector of L regressor terms, ϑ_k denotes the vector of parameter estimates calculated at the k-th time-instant and L is a positive user-defined integer denoting the size of the function approximator (3.11). The vector ϕ of regressor terms must be chosen so that it satisfies the so-called *Universal Approximation Property* [87], i.e. it must be chosen so that the approximation accuracy of the approximator (3.11) 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 (see [87] and the references therein).

The parameter estimation vector ϑ_k is calculated according to

$$\vartheta_k = \underset{\vartheta}{\operatorname{argmin}} \frac{1}{2} \sum_{\ell=\ell_k}^{k-1} \left(J_\ell^n - \vartheta^\tau \phi \left(x_\ell^{(1)}, \ldots, x_\ell^{(M)}, \bar{\theta}_\ell \right) \right)^2 \tag{3.12}$$

where $\ell_k = \max\{0, k - L - T_h\}$ with T_h being a user-defined nonnegative integer which defines the memory of the system. Standard least-squares optimization algorithms can be used to obtain a solution of (3.12).

To better understand the usage of the estimator (3.11) within the proposed algorithm, assume for the time-being that the sequence θ_k is exactly known (i.e. $\theta_k \equiv \bar{\theta}_k, \forall k$); then, as previously discussed, if the regressor vector ϕ is chosen to belong to a family of Universal

Approximators (e.g. polynomials, neural networks, etc) it can be seen using standard arguments (see e.g. [62, 48, 47, 75] and the references therein) that the choice for ϑ_k according to (3.12) guarantees that for any bounded $x \in \Re^{n_x}$,

$$J(x, \theta_k) = \hat{J}_k(x, \bar{\theta}_k) + \nu_k(x, \bar{\theta}_k)\,, \tag{3.13}$$

where ν_k is a term that can be made arbitrarily small (for k large enough) provided that:

- the "size" L of the regressor vector ϕ is large enough,
- the vector sequence

$$\phi_k \equiv \phi(x_k, \bar{\theta}_k)$$

satisfies a *Persistence of Excitation (PE)* condition, see e.g. [50]. Long-standing results in the theory of adaptive estimation and system identification (see e.g. [50] and the references therein) have established that PE is sufficient as well as necessary for the convergence of the parameter estimates ϑ_k to their optimal values; if a PE condition does not hold then the term ν_k in (3.13) may be particularly large no matter what the choice for L is. We close this parenthesis by noting that, since in the general case the exogenous vector θ_k is not exactly known, equation (3.13) should be replaced by:

$$J(x, \theta_k) = \hat{J}_k(x, \bar{\theta}_k) + \nu_k(x, \bar{\theta}_k) + \chi_1\left(\theta_k - \bar{\theta}_k\right)\,, \tag{3.14}$$

where $\chi_1 \in \mathsf{K}_\infty$. The proposed algorithm attempts to construct a sequence of system vectors x_k which guarantees, on the one hand, that the vector sequence ϕ_k satisfies a PE condition, and, on the other hand, that at each algorithm iteration the new system vector x_k leads to a non-negligible decrease of the estimate $\hat{J}_k$ (with respect to $\hat{J}_{k-1}$); if ϕ_k satisfies a PE condition then $\hat{J}_k$ will be an effective approximation of J_k – see eq. (3.14) – and thus the choice of x_k that leads to a non-negligible decrease of the estimate $\hat{J}_k$ is most likely to lead to a non-negligible decrease of J_k, too.

We are now ready to describe the procedure used by the proposed algorithm for choosing x_k: let α_k be a user-defined scalar positive sequence and

$$\Delta x_k^{(j)} \in \{-\alpha_k, +\alpha_k\}^{n_x},\ j \in \{1, \ldots, N\}$$

denote a collection of $N \geq n_x$ vectors of candidate perturbations, satisfying $\forall j \in \{1, \ldots, N\}$

$$\operatorname{rank}\left[\phi_{\max\{0,k-L+1\}}, \ldots, \phi_{k-1}, \phi(x_{k-1} \pm \Delta x_k^{(j)}, \bar{\theta}_k)\right] = \min\{k-1, L\} \tag{3.15}$$

and

$$\left|\left[\Delta x_k^{(1)},\ldots,\Delta x_k^{(K)}\right]^{-1}\right| \leq \frac{\Xi}{\alpha_k}\,, \tag{3.16}$$

where Ξ is a finite positive constant independent of α_k. Let also (here e_i is defined as $e_{ii}=1$ and $e_{ij}=0, j\neq i$)

$$\widehat{\nabla J}_k = \frac{\mathrm{vec}\left(\hat{J}_k(x_{k-1}+c_k e_i,\bar{\theta}_k)-\hat{J}_k(x_{k-1},\bar{\theta}_k)\right)}{c_k}\,, \tag{3.17}$$

where c_k is a user-defined positive scalar sequence. Then $\Delta x_k \equiv x_k - x_{k-1}$ is chosen according to:

$$\Delta x_k = \arg\min_{\pm\Delta x_k^{(j)}, j\in\{1,\ldots,N\}} \left(\pm\Delta x_k^{(j)}\right)^{\tau}\widehat{\nabla J}_k\,. \tag{3.18}$$

The key idea of the proposed algorithm is to use the approximation $\hat{J}_k$ to estimate the effect of the candidate perturbations $\pm\Delta x_k^{(j)}$ to the objective function and choose the perturbation that leads to the maximum (estimated) decrease of the objective function. This is realized in the proposed algorithm through the decision mechanism (3.18), which chooses one among the candidate perturbations $\pm\Delta x_k^{(j)}$. The motivation behind using (3.18) is that, if $\hat{J}_k$ as calculated in (3.11), (3.12) is an accurate estimate of J (i.e. $\hat{J}_k \approx J$), then $\widehat{\nabla J}_k \approx \nabla J$ in which case it is straightforward for someone to see that Δx_k generated according to (3.18) leads to a non-negligible decrease of the objective function.

Condition (3.15) is a standard Persistence of Excitation condition while condition (3.16) is imposed to make sure that there exists at least one candidate perturbation $\pm\Delta x_k^{(j)}$ that leads to a non-negligible decrease of J. Note that condition (3.15) renders the problem of finding $\Delta x_k^{(j)}$ a computationally hard problem. Fortunately, as we will see in Proposition 1, under appropriate selection of the regressor vector ϕ and the random generator for producing $\Delta x_k^{(j)}$, there is no practical need to check the computationally "heavy" condition (3.15).

For clarity's sake, let us come back to our particular coverage problem expressed by the objective function (3.1) and we include now also the set of constraints (3.3). In this case, if we assume that the optimization function is only dependent on the state vector x (i.e. the robots' positions), we can express in a more compact form the fundamental steps of the proposed algorithm. As soon as the estimator $\hat{J}_k$ is constructed according to (3.11), (3.12), the set of new robots' positions/poses is selected as follows: firstly, a set of N candidate robots'

positions/poses is constructed according to[1]

$$x_k^{i,j} = x_k^{(i)} + \alpha_k \zeta_k^{i,j}, i \in \{1, \ldots, M\}, j \in \{1, \ldots, N\}\,, \tag{3.19}$$

where $\zeta_k^{i,j}$ is a zero-mean, unity-variance random vector with dimension equal to the dimension of $x_k^{(i)}$ and α_k is a positive real sequence which satisfies the conditions:

$$\lim_{k\to\infty} \alpha_k = 0, \quad \sum_{k=1}^{\infty} \alpha_k = \infty, \quad \sum_{k=1}^{\infty} \alpha_k^2 < \infty\,. \tag{3.20}$$

Among all N candidate new positions $x_k^{1,j}, \ldots, x_k^{M,j}$, the ones that correspond to non-feasible positions/poses, i.e., the ones that violate the constraints (3.3), are projected onto the feasible space and then the new robots' positions/poses are calculated as follows:

$$\left[x_{k+1}^{(1)}, \ldots, x_{k+1}^{(M)}\right] = \underset{\substack{j \in \{1,\ldots,N\} \\ x_k^{i,j}\text{constrained}}}{\operatorname{argmin}} \hat{J}_k\left(x_k^{1,j}, \ldots, x_k^{M,j}\right)\,.$$

The idea behind the above logic is simple: at each time-instant a set of many candidate new robots' positions/poses is generated. The candidate, after the projection of those that do not provide with a feasible solution, that corresponds to the best estimated value $\hat{J}_k$ of the coverage criterion is selected as the new set of robots' positions/poses. The main steps of the CAO algorithm are summarized in Fig. 3.1. The random choice for the candidates is essential and crucial for the efficiency of the algorithm, as such a choice guarantees that $\hat{J}_k$ is a reliable and accurate estimate for the unknown function $\mathcal{J}$ (see [63, 60] for more details). On the other hand, the choice of a slowly decaying sequence α_k, a typical choice of adaptive gains in stochastic optimization algorithms (see e.g., [8]) is essential for filtering out the effects of the noise term ξ_k in eq. (3.2). The next theorem easily summarizes the properties of the constrained CAO algorithm described above, as it is employed in this thesis; the rigorous proof of this theorem is provided in the next section, after the proof of convergence for the unconstrained version of the algorithm.

Theorem 1 *Let $x^{(1^*)}, \ldots, x^{(M^*)}$ denote any – local – minimum of the constrained optimization problem (3.4). Let $N \geq 2M \times \dim\left(x_k^{(i)}\right)$ and, moreover, the vector ϕ satisfy the Universal Approximation Property. Assume also that the functions $\mathcal{J}, \mathcal{C}$ are either continuous or*

[1] According to [63, 60] 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/poses $x_k^{(1)}, \ldots, x_k^{(M)}$ and thus it suffices for N to satisfy $N \geq 2M \times \dim\left(x_k^{(i)}\right)$.

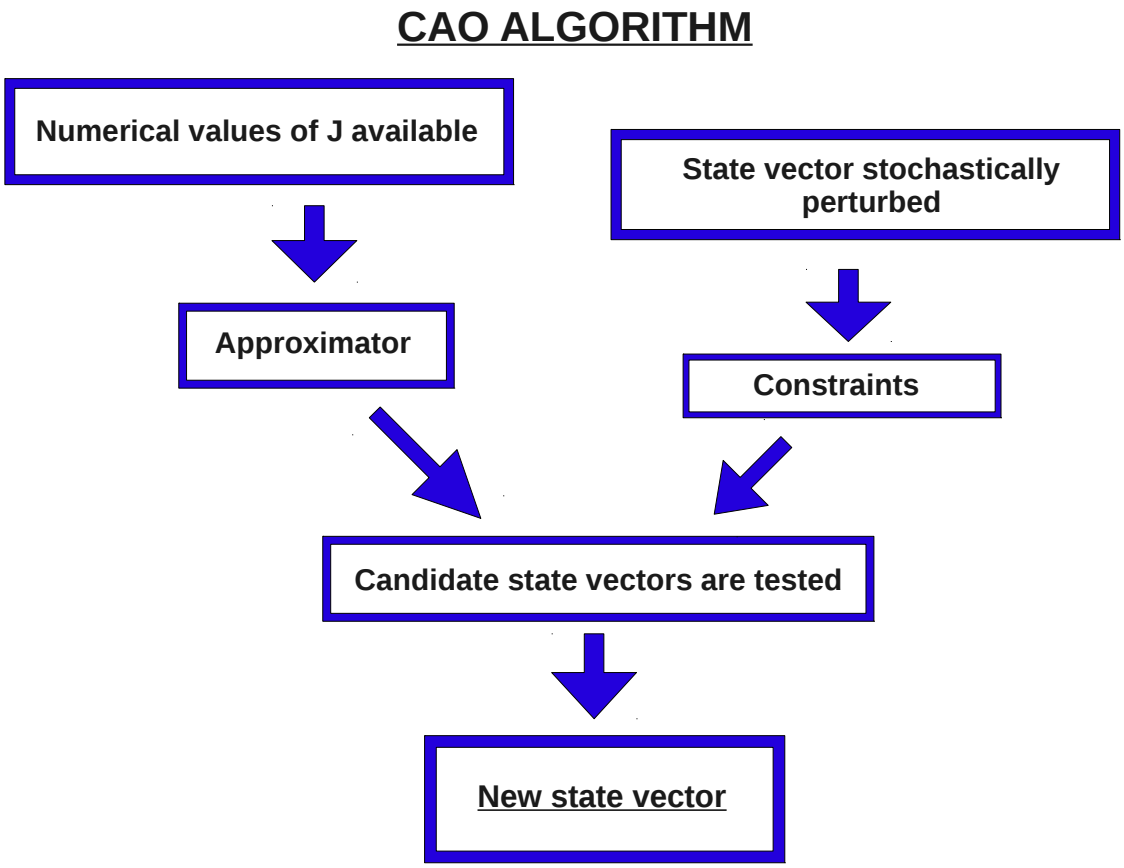


Figure 3.1: Main steps of the Cognitive-based Adaptive Optimization algorithm.

discontinuous with a finite number of[2] *discontinuities. Then, the CAO-based multi-robot coverage algorithm as described above guarantees that the robots' positions/poses* $x_k^{(1)}, \ldots, x_k^{(M)}$ *will converge to one of the local minima* $x^{(1^*)}, \ldots, x^{(M^*)}$ *almost surely, provided that the size* L *of the regressor vector* ϕ *is larger than a lower bound* $\bar{L}$*.*

Remark 4 [$\zeta_k^{i,j}$ vector] Strictly speaking, Theorem 1 is valid as long as the zero-mean, unity variance vectors $\zeta_k^{i,j}$ satisfy some extra technical conditions (which are satisfied if e.g., $\zeta_k^{i,j}$ are Bernoulli random vectors). However, extensive simulation investigations have shown that, in practice, Theorem 1 is still valid even if the random vectors $\zeta_k^{i,j}$ are Gaussian random vectors, despite the fact that such a choice does not satisfy the aforementioned technical conditions. ◇

Remark 5 [Lower bound for L] As already noticed, the CAO algorithm requires only a

[2]Note that the family of "discontinuous functions with a finite number discontinuities" corresponds to the family of functions that can be approximated with arbitrary accuracy by continuous ones [51]. For instance, terrains with discontinuities along e.g., a closed or open curve belong to this family of functions and so do the corresponding functions $\mathcal{J}$ and $\mathcal{C}$.

local approximation of the unknown function $\mathcal{J}$ and as a result the lower bound $\bar{L}$ has not to be large (as opposed to methods that construct a global approximation of the unknown function $\mathcal{J}$). Although, there exist no theoretical results for providing the lower bound $\bar{L}$ for the size of the regressor vector ϕ, practical investigations on many different problems indicate that for the choice of the regressor vectors such a bound is $2\times$[number of variables being optimized by CAO]; see [61, 63, 60] for more details. $\diamond$

Remark 6 [SPSA algorithm] As an alternative to the CAO approach, the SPSA approach [108] may be employed in multi-robot coverage applications. According to the SPSA approach, the robot positions/poses are updated according to

$$\begin{cases} x_{k+1}^{(i)} = x_k^{(i)} + \beta_k \zeta_k^i \,, & \text{if } k \text{ is even} \\ x_{k+1}^{(i)} = x_k^{(i)} + \gamma_k \frac{J_k^n - J_{k-1}^n}{\zeta_{k-1}^{(i)}} \,, & \text{if } k \text{ is odd} \end{cases} \tag{3.21}$$

where $\zeta_k^{(i)}$ are zero-mean, unity-variance random vectors and β_k, γ_k are slowly decaying sequences (similar as the sequence α_k). The SPSA algorithm is computationally simpler than the CAO one, but it does not perform as efficient as the CAO approach as have been demonstrated in a variety of approaches, see [61, 63, 60]. However, extensive simulation experiments have demonstrated that a hybrid scheme which uses SPSA at the first 10-20 time-steps and then switches to the CAO algorithm can have significant improvements over schemes that employ only the CAO algorithm. This is due to the fact that CAO, at its initial steps, may preserve a poor performance because it takes some iterations for the CAO estimator (3.11) in order to come up with a reliable estimate $\hat{J}_k$ of the unknown optimization function $\mathcal{J}$. $\diamond$

Remark 7 [Global optimization algorithms] We close this section by mentioning that similarly to the proposed approach, global optimization methods such as simulated annealing and genetic algorithms do not require that the explicit form of the function $\mathcal{J}$ is known. However, simulated annealing, genetic algorithms and other similar global optimization methods require that a large amount of different combinations of robots' positions is being evaluated all over the robots' application area. Such a requirement renders these methods practically infeasible as a huge amount of time and energy would have to be spent in order for the robots to visit many different locations all over their application area. Nevertheless, attempting to globally optimize surveillance coverage is practically infeasible as it is an NP-hard problem whose solution requires dense discretization over the space of all possible team configurations and evaluation of all points of the discretized space. $\diamond$

3.3 Convergence Properties

We present here the rigorous enunciation and proof of the convergence properties of the CAO algorithm. The first main result establishes satisfactory transient performance and convergence of the proposed algorithm in the case of (uniformly bounded) deterministic exogenous signals for an unconstrained problem. The two following theorems, Theorems 2 and 3, have been presented by Kosmatopoulos in [63]. As we explain at the end of this section, the proof for constrained problem, i.e. the proof of Theorem 1, is an extension of the following results.

Theorem 2 *Let*

$$\vartheta^* = \arg\min_{\vartheta} \sup_{\theta:|\theta|\leq c_\theta, x:|x|\leq c_x} |J(x,\theta) - \vartheta^\tau \phi(x,\theta)|$$

and $\nu(x,\theta) = J(x,\theta) - \vartheta^{*\tau}\phi(x,\theta)$*,*

$$\underline{\nu} = \sup_{\theta:|\theta|\leq c_\theta, x:|x|\leq c_x} |J(x,\theta) - \vartheta^\tau \phi(x,\theta)|$$

and suppose that there exist finite nonnegative constants $c_\theta, c_{\bar\theta}, c_{\Delta\theta}, c_x$ *such that, for each* $k \in Z_+$*, the following hold:*

(A1) $|\theta_k| < c_\theta, |\theta_k - \bar\theta_k| < c_{\bar\theta}, |\theta_k - \theta_{k-1}| < c_{\Delta\theta}$.

(A2) *The proposed algorithm admits a solution satisfying (3.15), (3.16).*

(A3) *The proposed algorithm guarantees that* $|x| \leq c_x$.

(A4) *The user-defined sequences* α_k, c_k *satisfy* $\alpha_k \geq \alpha > 0$, $\bar c > c_k \geq \underline{c} > 0$.

Then, the proposed algorithm is a $(\bar\sigma, \bar\tau_k)$-Adaptive Optimization algorithm *with*

$$\bar\sigma = \frac{1}{\alpha}\max\{\delta_1, \delta_2\}, \ \bar\tau_k = \begin{cases} 0 & \text{if } |\nabla J(x_{k-1}, \theta_k)| > \varepsilon_k \\ \delta_{3,k} & \text{otherwise} \end{cases}$$

where:

$$\delta_1 = \bar c O(1) + \chi_2(\underline{\nu}) + \chi_3(c_{\bar\theta}),$$

$$\delta_2 = \alpha^2 O(1) + \chi_4(c_{\Delta\theta}),$$

$$\varepsilon_k = \frac{1}{\alpha_k}\max\{c_k O(1) + \chi_2(\underline{\nu}) + \chi_5(\eta_k) + \chi_3(\left|\theta_{\{\ell_k,\ldots,k\}} - \bar\theta_{\{\ell_k,\ldots,k\}}\right|),$$

$$\alpha_k^2 O(1) + \chi_4(|\theta_k - \theta_{k-1}|)\},$$

$$\delta_{3,k} = \alpha_k\varepsilon_k + \chi_4(|\theta_k - \theta_{k-1}|) + \chi_3(|\theta_{\{\ell_k,\ldots,k\}} - \bar\theta_{\{\ell_k,\ldots,k\}}|)\alpha_k^2,$$

$$\chi_i, \quad i = 2,\ldots,5 \in \mathsf{K}_\infty$$

and η_k *is a bounded term satisfying* $\eta_k = 0, \forall k \geq L$.

Proof. Let $\Pi_k \equiv \nabla J(x_{k-1}, \theta_k)$ and note that

$$J(x,\theta) = (\vartheta^*)^\tau \, \phi(x,\bar{\theta}) + \bar{\nu}(x,\theta,\bar{\theta}) \,, \tag{3.22}$$

where

$$\bar{\nu}(x,\theta,\bar{\theta}) = (\vartheta^*)^\tau \left[\phi(x,\theta) - \phi(x,\bar{\theta})\right] + \nu(x,\theta) \,. \tag{3.23}$$

Using condition (3.15) it can be seen that the solution $\hat{J}_k$ calculated according to (3.11), (3.12) satisfies

$$J(x,\theta) = \hat{J}_k(x,\bar{\theta}) + \mu_k(x,\theta,\bar{\theta}) \,, \tag{3.24}$$

where

$$\mu_k(x,\theta,\bar{\theta}) = \bar{\bar{\nu}}_k \phi(x,\bar{\theta}) + \bar{\nu}(x,\theta,\bar{\theta}) + \eta_k^\tau \phi(x,\bar{\theta})$$

with

$$\bar{\bar{\nu}}_k = O\left(\bar{\nu}(\theta_{\{\ell_k,\ldots,k\}}, \theta_{\{\ell_k,\ldots,k\}}, \bar{\theta}_{\{\ell_k,\ldots,k\}})\right)$$

and $\eta_k = 0, \forall k \geq L$; using (3.24) we directly obtain that:

$$\begin{aligned}
&\frac{\partial}{\partial x_i} J(x,\theta) - \frac{1}{c_k}\hat{J}_k(x + c_k e_i, \bar{\theta}) + \frac{1}{c_k}\hat{J}_k(x,\bar{\theta}) \\
&= \int_0^1 \left(\frac{\partial}{\partial x_i} J(x,\theta) - \frac{\partial}{\partial x_i} J(x + s c_k e_i, \theta)\right) ds \\
&+ \frac{1}{c_k}(\mu_k(x + c_k e_i, \theta, \bar{\theta}) - \mu_k(x,\theta,\bar{\theta})) \equiv \epsilon_{k,i}(x,\theta,\bar{\theta}) \,.
\end{aligned} \tag{3.25}$$

Using this equality and (3.18), we readily obtain that:

$$\Delta x_k^\tau \Pi_k = \arg \min_{\pm \Delta x_k^{(j)}} [(\pm \Delta x_k^{(j)})^\tau (\Pi_k - \epsilon_k)]^\tau \Pi_k \equiv H_k \,, \tag{3.26}$$

where $\epsilon_k = \mathrm{vec}(\epsilon_{k,i}(x_{k-1}, \theta_k, \bar{\theta}_k)))$, which implies that there exists a $\bar{\chi}_1 \in \mathsf{K}_\infty$ such that:

$$\begin{aligned}
&|\Pi_k| > \bar{\epsilon}_k \equiv \bar{\chi}_1 \left(\max_i \left|\mathrm{vec}\left(\epsilon_{k,i}(x_{k-1}, \theta_k, \bar{\theta}_k)\right)\right|\right) \\
&\Rightarrow H_k = \min_{\pm \Delta x_k^{(j)}} \left[\left(\pm \Delta x_k^{(j)}\right)^\tau \Pi_k\right] \equiv \bar{H}_k \leq -\frac{\alpha_k}{\Xi n_x} |\Pi_k| \,,
\end{aligned} \tag{3.27}$$

where the last inequality was obtained by using (3.16). Therefore, using standard Taylor expansion arguments (see e.g. Prop. 1, [8]) it can be seen that $|\Pi_k| > \bar{\epsilon}_k$

$$\begin{aligned} &\Rightarrow J_k \leq J_{k-1} + \Delta x_k^{\tau}\Pi_k + \gamma_1(\theta_{k-1})n_x\alpha_k^2 + \gamma_2(|\theta_k, \theta_{k-1}|) \\ &\leq J_{k-1} - \frac{\alpha_k}{\Xi n_x}|\Pi_k| + \gamma_1(\theta_{k-1})n_x\alpha_k^2 + \gamma_2(|\theta_k - \theta_{k-1}|) \end{aligned} \tag{3.28}$$

for some positive function γ_1 and $\gamma_2 \in \mathsf{K}_\infty$, or, equivalently,

$$\begin{aligned} J_k \;\leq\; & J_{k-1} - I_k\frac{\alpha_k}{\Xi n_\theta}|\Pi_k| + (1 - I_k)\,\bar{\epsilon}_k\alpha_k\sqrt{n_x} \\ & + \gamma_1(\theta_{k-1})n_x\alpha_k^2 + \gamma_2(|\theta_k - \theta_{k-1}|) \end{aligned} \tag{3.29}$$

where I_k is defined as $I_k = 1$ if $|\Pi_k| > \bar{\epsilon}_k$ and $I_k = 0$, otherwise. Using the above inequality together with (A1), the fact that J is at-least C^2 and the definition of $\bar{\epsilon}_k, \mu_k$ we can readily establish the proof after some lengthy but quite straightforward calculations. $\triangle$

Roughly speaking, Theorem 2 states that, in the case of deterministic exogenous disturbances and without constraints, the terms $\bar{\sigma}$ and $\bar{\tau}_k$ are, in the worst case, proportional to:
(a) the approximation accuracy ν of the function approximator (3.11), which can be made arbitrarily small (see Proposition 1),
(b) the estimation accuracy $\theta_k - \bar{\theta}_k$,
(c) the exogenous signals' "velocity" $\theta_k - \theta_{k-1}$.

Additionally, the term $\bar{\tau}_k$ is also affected by an extra term (the term η_k) which becomes negligible for $k \geq L$; the presence of η_k is unavoidable since, initially, the function J is totally unknown and it takes some iterations for the proposed algorithm to produce an effective estimate of this function. Next we present some comments regarding assumptions (A1)-(A4).

Remark 8 [Assumptions (A1)-(A4)] Assumption (A1) requires that the exogenous signal θ_k is uniformly bounded; note that the proposed algorithm does not require knowledge of the bounds $c_\theta, c_{\bar{\theta}}, c_{\Delta\theta}$. Assumption (A2) is quite difficult to verify for a general choice of the regressor vector ϕ_k; however, as we establish in Proposition 1, if ϕ_k is chosen to be either a polynomial or a neural network of a specific structure, then assumption (A2) is trivially satisfied if the candidate perturbations $\Delta x_k^{(j)}$ are Bernoulli-like random terms. Assumption (A3) is imposed in order to avoid lengthy technicalities in the presented proof. It is not difficult for someone to see that all of the results of this chapter are valid if we remove assumption (A3) and use a projection mechanism as in [65] for keeping x_k bounded; similarly to [65] it can be seen that the introduction of such mechanisms does not destroy the performance and convergence properties of the proposed algorithm. Finally, assumption (A4) is a standard SA assumption on updating schemes with fixed step-sizes (see e.g. [14]).

The next Theorem establishes the properties of the proposed algorithm if we remove assumption (A1) on boundedness of the exogenous signal θ_k and assume instead that θ_k and $\theta_k - \bar{\theta}_k$ are random vectors (not necessarily uniformly bounded) that are zero-mean with finite variance (with respect to an appropriately defined σ-field). Note that in this case the sequences α_k, c_k should be chosen to be slowly decaying to zero terms.

Theorem 3 *Suppose that (A2)-(A3) hold and additionally that the following assumptions hold:*

(A1′) $E\left[\theta_k - \bar{\theta}_k | G_{k-1}\right] = 0, E\left[\theta_k | G_{k-1}\right] = 0$, $E\left[|\theta_k|^2 | G_{k-1}\right] < \infty$, *where* G_k *denotes the* σ*-field generated by* $\{\theta_0, \ldots, \theta_k, \bar{\theta}_0, \ldots, \bar{\theta}_k, \Delta x_0^{(j)}, \ldots, \Delta x_k^{(j)}\}$.

(A4′) *The user-defined sequences* α_k, c_k *satisfy*

$$\lim_{k\to\infty} \alpha_k = 0, \quad \sum_{k=0}^{\infty} \alpha_k = \infty, \quad \sum_{k=0}^{\infty} \alpha_k^2 < \infty,$$
$$\sum_{k=0}^{\infty} \alpha_k c_k < \infty, \quad \lim_{k\to\infty} \alpha_k / c_k = 0\,. \tag{3.30}$$

Suppose moreover that T_h *is chosen according* $T_h = k - \bar{L}$ *where* $\bar{L}$ *is*[3] *any positive integer. Finally, let* $\tilde{\vartheta}_k^*, \tilde{\nu}_k$ *be defined according to*

$$\tilde{\vartheta}_k^* = \arg\min_{\vartheta} E\left[|J(x_k, \theta_k) - \vartheta^\tau \phi(x_k, \theta_k)|^2 | G_{k-1}\right]$$

$$\tilde{\nu}_k(x, \theta) = \sup_{\ell \in \{\ell_k, \ldots, k\}} \left| J(x_\ell, \theta_\ell) - \left(\tilde{\vartheta}_k^*\right)^\tau \phi(x_\ell, \theta_\ell) \right|$$

Then, the proposed algorithm is a $(\tilde{\sigma}, \tilde{\tau}_k)$ - Adaptive Optimization algorithm *with*

$$\tilde{\sigma} = E\left[\chi_6\left(\tilde{\nu}_k\right) | G_{k-1}\right], \quad \chi_6 \in \mathsf{K}_\infty \tag{3.31}$$

and $\tilde{\tau}_k$ *is defined as* $\bar{\tau}_k$ *in Theorem 2 by replacing* α_k *by some positive constant* $\alpha \in (0, \alpha_0)$ *and* ν *by* $\tilde{\nu}_k$.

Proof. The analysis of the proof of Theorem 2 leading to inequality (3.29) is valid here as well (by replacing ϑ^*, ν in the proof of Theorem 2 by $\tilde{\vartheta}_k^*, \tilde{\nu}_k$ defined in Theorem 3, respectively). Taking conditional expectations on (3.29) and using (A1′) and some lengthy, but quite

[3] Due the fact that $\alpha_k \mapsto 0$, it is necessary to impose condition $T_h = k - \bar{L}$; if this condition does not hold then (A2) will not hold as well since (3.15) will not admit a bounded solution for $k \mapsto \infty$.

straightforward calculations, we obtain that (here, for notational convenience we use the notation $E_{k-1}[\,\cdot\,] \equiv E[\,\cdot\,|G_{k-1}]$)

$$\begin{aligned} & E_{k-1}\left[J_k\right] \\ \leq\ & J_{k-1} - E_{k-1}\left[I_k \frac{\alpha_k}{\Xi n_x}\left|\Pi_k\right|\right] + \bar{\gamma}_1 \alpha_k^2 + E_{k-1}\left[\left(1-I_k\right)\tilde{\epsilon}_k \alpha_k\right] \\ \leq\ & J_{k-1} - X_k + \bar{\gamma}_1 \alpha_k^2 + E_{k-1}\left[\left(1-I_k\right)\right]\alpha_k c_k O(1) \\ & + E_{k-1}\left[\left(1-I_k\right)\alpha_k\left(\psi_k(\theta_{\{\ell_k,\ldots k\}}) - \psi_k(\bar{\theta}_{\{\ell_k,\ldots k\}})\right)\right] \end{aligned} \tag{3.32}$$

where (note that the term $\bar{\gamma}_1$ is bounded since, from (A1′), $E_{k-1}\left[|\theta_k|^2\right] < \infty$)

$$\bar{\gamma}_1 = n_\theta E_{k-1}\left[\gamma_1(\theta_k)\right]\,,$$

the term $\tilde{\epsilon}_k$ is defined similar to the term $\bar{\epsilon}_k$ in the proof of Theorem 2, $\bar{\chi}_2 \in \mathsf{K}_\infty$, $\psi_k(\cdot)$ is an appropriately defined function (that depends on $\phi(\cdot,\cdot)$) and $X_k = Y_k$ if $Y_k \geq 0$ and $X_k = 0$, otherwise, with

$$Y_k = E_{k-1}\left[I_k \frac{\alpha_k}{\Xi n_x}\left|\Pi_k\right| - \left(1-I_k\right)\alpha_k \bar{\chi}_2(\tilde{\nu}_k)\right]\,. \tag{3.33}$$

Assumption (A4′) guarantees that $\sum_{k=0}^{\infty} \bar{\gamma}_1 \alpha_k^2$ and $\sum_{k=1}^{\infty} E_{k-1}\left[\left(1-I_k\right)\right]\alpha_k c_k O(1)$ converge; moreover, using $E_{k-1}[\theta_k - \bar{\theta}_k] = 0$ (see assumption (A1′)), we have that:

$$E_{k-1}\left[\left(1-I_k\right)\alpha_k\left(\psi_k(\theta_{\{\ell_k,\ldots k\}}) - \psi_k(\bar{\theta}_{\{\ell_k,\ldots k\}})\right)\right] = 0\,. \tag{3.34}$$

Therefore, application of Robbins and Siegmud theorem [100] on nonnegative almost-supermartingales to inequality (3.32), establishes that $\sum_{k=0}^{\infty} X_k$ converges. Standard arguments can be now applied (see e.g. proof of part (b) of Proposition 3.1 of [22]) to show that the convergence of $\sum_k X_k$ together with the facts that $\sum_{k=0}^{\infty} \alpha_k = \infty$ and ∇J is at least C^1 imply

$$\lim_{k\to\infty} \frac{1}{\alpha_k} X_k = 0\,,$$

or, equivalently that

$$\lim_{k\to\infty} \frac{1}{\alpha_k} Y_k \leq 0\,;$$

the last inequality implies that

$$\lim_{k\to\infty} E_{k-1}\left[\left(1-I_k\right)\right] < 1$$

unless $\tilde{\nu}_k \to 0$, and thus

$$\lim_{k\to\infty} E_{k-1}\left[I_k\right] > 0\,,$$

which, in turn, implies that for

$$k \mapsto \infty, \quad E_{k-1}\left[\frac{1}{\Xi n_x}\left|\Pi_k\right|\right] \leq E_{k-1}\left[\bar{\chi}_2(\tilde{\nu}_k)\right]$$

which establishes (3.31). The rest of the proof is similar to that of Theorem 2. △

Theorem 3 establishes that in the case θ_k is stochastic vector sequence satisfying (A1′) then J_k preserves similar transient performance properties as in the case of deterministic sequences; moreover Theorem 3 establishes that the parameter vector sequence x_k converges (with probability 1) arbitrarily close to a local minimum of J (under appropriate selection of the approximator (3.11) – see Proposition 1 below).

Remark 9 [Assumptions (A1′), (A2′)] Assumption (A1′) is also a quite standard assumption in SA (see e.g. [8]). Assumption (A2′) is a standard assumption on SA algorithms with vanishing gains.

Theorem 1 [Constrained optimization problem] - Proof. So far, we have proved the convergence properties of the CAO algorithm only for its unconstrained version. This was just for simplicity's sake and the extension for the constrained algorithm, used in this thesis, is now straight. Hence, the aim is to provide a proof of the Theorem 1, which is a contribution of this thesis, presented in the previous section.

Let x_k denotes the augmented vector of all robots' positions/poses at time-instant k, i.e., the entries of x_k are the entries of all vectors $x_k^{(1)}, \ldots, x_k^{(M)}$. Using similar arguments as those of the previous theorems, it can be seen that at each iteration of the CAO-based algorithm described above, the new vector x_{k+1} satisfies:

$$x_{k+1} = \Upsilon_C\left\{x_k - \alpha_k\left(c\nabla J(x_k) + e_k + b_k\right)\right\}\,, \tag{3.35}$$

where c is a positive constant; $\Upsilon_C\{\cdot\}$ denotes the projection operator onto the set $S = \{X : \mathcal{C}\left(X\right) \leq 0\}$ defined as follows: for any x not satisfying the constraint $\mathcal{C}\left(X\right) \leq 0$, the point $\tilde{x} = \Pi_C\{x\}$ is the nearest point to x on S, where the norm is defined in the usual Euclidean norm; and e_k, b_k are two terms that are defined similarly to the respective terms in section III of [108]. This extension is similar to that one for the constrained SPSA algorithm presented by Sadegh in [102] (see section 2.3.4). By using the same arguments, as those in the proof of Proposition 1 of [102], it can be established that the above equation converges almost surely to one of the local minima of the constraint minimization problem (3.4).

In the next proposition we show that if the estimator (3.11) is chosen to be either an Incremental-Extreme Learning Machine (I-ELM) [48, 47] or a Polynomial-Like Universal Approximator (PLUA) [62] and, moreover, the candidate perturbations $\Delta\theta_k^{(j)}$ are Bernoulli-like random terms, then there is no need to check the computationally heavy condition (3.15); moreover, Proposition 1 establishes that the terms $\nu, E\left[\chi_6\left(\tilde{\nu}_k\right)|G_{k-1}\right]$ defined in Theorems 2 and 3, respectively, can be made arbitrarily small by increasing the "size" L of the estimator (3.11).

Proposition 1 *Suppose that the regressor vector ϕ is selected according to one of the following:*
(I-ELM) $\phi_i(x,\theta) = S(A_i^\tau \mathrm{vec}(x,\theta) + b_i), i \in \{1,\dots,L\}$, *where $S(\cdot)$ is an invertible smooth nonlinear function and the vectors A_i and the real parameters b_i are randomly generated (with A_i, b_i being zero-mean), or*
(PLUA) $\phi_i(x,\theta) = S(x_1)^{d_{i,1}^x}\dots S(x_{n_x})^{d_{i,n_x}^x}\bar{S}(\theta_1)^{d_{i,1}^\theta}\dots\bar{S}(\theta_{n_\theta})^{d_{i,n_\theta}^\theta}$, $i \in \{1,\dots,L\}$, *where S is any smooth monotone function and $d_{i,j}^x, d_{i,j}^\theta$ are nonnegative integers such that*

$$\bar{S}(\bar{\theta}_{k,1})^{d_{i,1}^\theta}\dots\bar{S}(\bar{\theta}_{k,n_\theta})^{d_{i,n_\theta}^\theta} \neq 0,\ \forall k,i$$

and moreover the integers $d_{i,j}^x$ are such that

$$\exists j \in \{1,\dots,n_x\} : d_{i,j}^x > 0,\ \forall i \in \{1,\dots,L\}\,.$$

Moreover assume that $\Delta x_k^{(j)}$ are random zero-mean vectors in $\{-\alpha_k,+\alpha_k\}^{n_x}$ satisfying[4] *(3.16). Then, condition (3.15) is satisfied with probability 1. Moreover, the terms $\nu(\cdot)$ and $E\left[\chi_6\left(\tilde{\nu}_k\right)|G_{k-1}\right]$ defined in Theorems 2 and 3, respectively, satisfy for $\chi_7, \chi_8 \in \mathsf{K}_\infty$*

$$\nu = \chi_7\left(1/L\right), \quad E\left[\chi_6\left(\tilde{\nu}_k\right)|G_{k-1}\right] = \chi_8\left(1/L\right) \tag{3.36}$$

Proof. We provide with a sketch of the proof only for the I-ELM case; the proof for the PLUA case is similar. Since S is invertible, if (3.15) does not hold then there exists a nonzero vector b such that

$$b^\tau\left(A\,\mathrm{vec}\left(\Delta x_\ell, \bar{\theta}_\ell\right) + b\right) = 0, \qquad \ell \in \{k-L_k+1,\dots,k-1\},$$

$$b^\tau\left(A\,\mathrm{vec}\left(\pm\Delta x_k^{(j)}, \bar{\theta}_k\right) + b\right) = 0\,,$$

[4] A choice $\Delta x_k^{(j)} = \alpha_k\Delta_k^{(j)}$ where $\Delta_k^{(j)}$ are Bernoulli random vectors satisfies (3.16); see also [9] for construction of zero-mean random or random-like sequences that satisfy condition (3.16).

where A denotes the matrix whose rows are the vectors A_i^τ and $b = \mathrm{vec}(b_i)$. Since A_i, b_i and $\Delta x_k^{(j)}$ are randomly chosen, it is quite straightforward to show that the probability a nonzero vector b to satisfy the above system of equations is zero. The proof of the first equation in (3.36) is based on standard approximation results over compact spaces (see e.g. [48] for the case of I-ELM and [62] for the case of PLUA), while the proof of the second equation in (3.36) can be obtained using the approximation results over unbounded spaces developed in [75]. △

We close this section by introducing some further remarks regarding the choice of the proposed algorithm's design parameters:

• Contrary to other applications of function approximators where the size L of an approximator of the form (3.11) should be significantly large to guarantee that it can approximate nonlinear functions over the whole input space, this is not the case here: in the case of the proposed algorithm it is sufficient that the approximator has enough regressor terms to come up with an approximation of the unknown function J over a small neighborhood around the most recent vector x_k.

• Having the above in mind, a relatively small (as compared to other applications of function approximators) number L of regressor terms should suffice for efficient algorithm performance; similarly, since the approximation required is over a small neighborhood of the current value of x_k a small time-window (determined by the parameter T_h in (3.12)) should be chosen. As a matter of fact, in all practical applications of algorithms using functions approximators for optimization purposes, (see [61]) as well as in various applications where we tested the proposed algorithm, a choice for L, T_h according to $L \approx 1/2(n_x + n_\theta), T_h = 50$ was found to produce quite satisfactory results. Moreover, in the case where a polynomial approximator is used, we found that it suffices to use a polynomial approximator of maximum order equal to 3 with randomly chosen polynomial terms at each algorithm iteration (i.e. $\sum_j d_{i,j}^x + \sum_\ell d_{i,\ell}^\theta = 3$ with $d_{i,j}^x, d_{i,j}^\theta$ randomly chosen).

• Finally, for the choice of the step-sizes α_k, c_k similar rules as the ones apply in standard SA algorithms can be used.

3.4 Conclusions

In this chapter the CAO algorithm has been presented. This new stochastic optimization method will be used to solve all the main problems considered in this thesis. For this reason an accurate description of all the steps and a mathematically rigorous proof of its convergence properties has been provided. In the following chapters we will present and analyze the differ-

ent possible coverage criteria, which are the objective functions the CAO algorithm is called to optimize. In particular, in the next chapter we consider the easier case of a 2D region and we show the first simulations results obtained by using the proposed method. We finally emphasized that this algorithm, as it has been possible to see from the description provided in this chapter, can be applied to a very wide class of problems. The main problem tackled in this thesis, the optimal surveillance coverage, is only one of the several possible applications and in chapter 7 we will prove this statement approaching a completely different robotic problem with the same method.

Chapter 4

Optimal Surveillance Coverage - 2D

The goal of this chapter is to apply the stochastic optimization algorithm presented in the previous chapter to the main problem tackled in this thesis: the optimal cooperative surveillance coverage. Even if the ultimate objective is to develop a strategy to deploy a swarm of micro aerial vehicles in a real environment, we start our analysis from the simpler case of a 2D area. This choice is essentially due to two reasons: the first one is that is easier starting our analysis with a simpler case, well studied in literature; secondly, it important to notice that a solution for 2D region can be convenient not only for a team of ground robots but also for a swarm of MAVs, if the terrain is sufficiently flat to be approximated with a plane.

First of all, we have to define what it is the meaning of optimizing the cooperative surveillance coverage in a given region. Two different criteria may be identified to answer to this question:

(O1) the part of the terrain that is monitored (i.e., is visible) by the robots must be maximized;

(O2) for every point in the terrain, the closest robot must be as close as possible to that point.

The first objective is the most intuitive in a surveillance task: knowing the positions from which it is possible to see as more as possible, regarding the sensors capabilities of the team. The second objective is may be necessary for two practical reasons: (a) firstly, in many multi-robot coverage applications there is the necessity of being able to intervene as fast as possible in any of the points of the terrain with at least one robot and (b) secondly, the closer is the robot to a point in the terrain the better is, in general, its sensing ability to monitor this point.

Of course, once the coverage criterion is well defined, the other consequent problem to solve is: given the initial robots' positions, finding feasible and safety trajectories to reach the desired final configuration. Moreover, in many practical application, the two problems are not separate but such trajectories should be designed in real-time, while the optimization problem is still to solve.

We firstly present the second criterion, which is the most studied in literature, showing some existing solutions and proposing a simple strategy to deal with the complex case of a non-convex area to cover. Then, we describe the first objective, more important for this thesis and we show how it is possible to apply the proposed stochastic optimization algorithm to find a solution of the problem. Additionally, as an important contribution of this chapter, after presenting the standard centralized solution, we propose also a distributed version to cooperatively achieve a maximum coverage. Finally we provide several numerical simulations to evaluate the performance of the algorithms.

4.1 Voronoi Coverage Control

The second objective, which we can identify as the intervention problem, has been extensively studied in literature and it leads, in the case of a non-convex 2D area, to the well known solution of the Voronoi coverage control. This problem was presented for the first time by Cortés *et al.* in [20]. This particular problem is not the main topic of this chapter but, for completeness, we describe the problem and the solution available in literature. Additionally, in section 4.2, we propose a simple and distributed solution for the more complex case of a non-convex and unknown environment.

4.1.1 Voronoi tessellation

Given an open set $\Omega \subseteq \mathbb{R}^N$ and a set of points $\mathcal{P}\{p_i\}_{i=1}^k$ belonging to Ω, the Voronoi region V_i corresponding to the point p_i is defined by

$$V_i = \{q \in \Omega \mid \quad \|q - p_i\| \leq \|q - p_j\| \ \forall j \neq i\}\,, \tag{4.1}$$

where $\|\cdot\|$ denote the Euclidean norm on $\mathbb{R}^N$. The points $\{p_i\}$ are called generators and the set $\{V_i\}_{i=1}^k$ is called the Voronoi tessellation (or partition) of Ω.

Given a region V and a density function $\phi(q)$, defined in V, the mass, the centroid (or center of mass) and the polar moment of inertia are defined as:

$$\begin{aligned} M_V &= \int_V \phi(q)dq\,, \quad C_V = \frac{1}{M_V}\int_V q\phi(q)dq\,, \\ J_{V,p} &= \int_V \|q-p\|^2\phi(q)dq\,. \end{aligned} \tag{4.2}$$

Additionally, we can write the relation between $J_{V,p}$ and the polar moment of inertia about the

center of mass, J_{V,C_V}:

$$J_{V,p} = J_{V,C_V} + M_V \|p - C_V\|^2 . \tag{4.3}$$

When the generators coincide with the centers of mass of their respective regions, the tessellation is known as *centroidal tessellation*. We address the reader to to [82] for a more extended treatment on Voronoi diagrams.

4.1.2 Problem formulation

To express the problem as an optimization problem, the first step is to define a function that quantifies the concept previously introduced and measures the degree of covering of the multi-robot team. Let p_i the position of the $i-th$ robot and $\mathcal{W}$ a tessellation of the region to cover, such that in each region W_i there is exactly one robot. Let us consider the function

$$J(\mathcal{P}, \mathcal{W}) = \sum_i \int_{W_i} f(\|q - p_i\|)\phi(q)dq , \tag{4.4}$$

where f is a monotonic differentiable function. The physical interpretation of this expression is the following: each point in W_i is weighted by

- f, which is a function of the distance between the point and the robot belonging to that region;
- ϕ, which gives information about the importance of the point.

Then, an integration over the whole region is made. The explicit dependence of f on the distance depends on the objective that we want to achieve. The final goal might be, for example, an optimal placement in order to minimize the time of intervention of at least one robot in some point of the space, or to maximize the information about the environment obtained by the robots sensors. In the first case, f will be an increasing function of the distance, in the second one it will depend on the specific characteristics of the sensor, but certainly decreasing with the distance. Our aim is to find the optimal partition $\mathcal{W}$ and optimal location $\mathcal{P}$ that extremize (minimize or maximize depending on the choice of the function $f(\|\cdot\|)$) $J(\mathcal{P}, \mathcal{W})$. It is possible to prove that, at a fixed robots location, the optimal partition is the Voronoi partition $\mathcal{V}$ [20]:

$$\min_{\mathcal{P},\mathcal{W}} J(\mathcal{P}, \mathcal{W}) = \min_{\mathcal{P}} J(\mathcal{P}, \mathcal{V}) . \tag{4.5}$$

Hence, we have to solve the optimization problem with respect to the location only, solving the equations:

$$\nabla J_{\mathcal{V}} = [\cdots \frac{\partial J_{\mathcal{V}}}{\partial p_i} \cdots]^T = 0 \tag{4.6}$$

where, for simplicity's sake, we have defined

$$J_{\mathcal{V}} \equiv J(\mathcal{P}, \mathcal{V}) \,. \tag{4.7}$$

The explicit expression of the components of (4.6) is:

$$\begin{aligned} \frac{\partial J_{\mathcal{V}}}{\partial p_i} = & \int_{V_i} \frac{\partial f(\|q - p_i\|)}{\partial p_i} \, \phi(q) \, dq + \sum_{j \in \mathcal{N}_i} \int_{\partial V_j} f(\|q - p_i\|) \, \phi(q) \frac{\partial \partial V_j}{\partial p_i} n_j \, dq \\ & + \int_{\partial V_i} f(\|q - p_i\|) \, \phi(q) \frac{\partial \partial V_i}{\partial p_i} n_i \, dq \end{aligned} \tag{4.8}$$

where ∂V_i denotes the boundary of the Voronoi region V_i, n_i denotes the outward facing normal of ∂V_i and $\mathcal{N}_i$ is the set of indices of the neighbors of p_i. It is possible to prove that the last two terms in (4.8) sum to zero. Since only the part of ∂V_j which is shared with the boundary of the region i gives contribution in (4.8), we can consider only these and thus we can write:

$$\bigcup_{j \in \mathcal{N}_i} \partial V_j = \partial V_i \,. \tag{4.9}$$

An inward normal $-n_i$ for V_i is equal to an outward normal n_j for any of its neighbors V_j, at the boundary which they share. This leads to

$$\sum_{j \in \mathcal{N}_i} \int_{\partial V_j} f(\|q - p_i\|) \, \phi(q) \frac{\partial \partial V_j}{\partial p_i} n_j \, dq = - \int_{\partial V_i} f(\|q - p_i\|) \, \phi(q) \frac{\partial \partial V_i}{\partial p_i} n_i \, dq \,. \tag{4.10}$$

Hence, we can write:

$$\frac{\partial J_{\mathcal{V}}}{\partial p_i} = \int_{V_i} \frac{\partial f(\|q - p_i\|)}{\partial p_i} \, \phi(q) \, dq = - \int_{V_i} \frac{df(x)}{dx}\Big|_{\|q - p_i\|} \frac{q - p_i}{\|q - p_i\|} \, \phi(q) \, dq \,. \tag{4.11}$$

Hereafter we consider only the unweighted problem, i.e. we fix $\phi(q) = 1$ and we restrict our attention to the case in which the region to cover is 2D.

We can obtain the explicit solution of the optimization problem for a particular choice of the function f:

$$f(\|q - p_i\|) = \|q - p_i\|^2 \,. \tag{4.12}$$

Indeed, in this case, the solution of the problem is very simple and the optimal location is the centroidal one, i.e. the robots are on the centers of mass of the respective regions. This

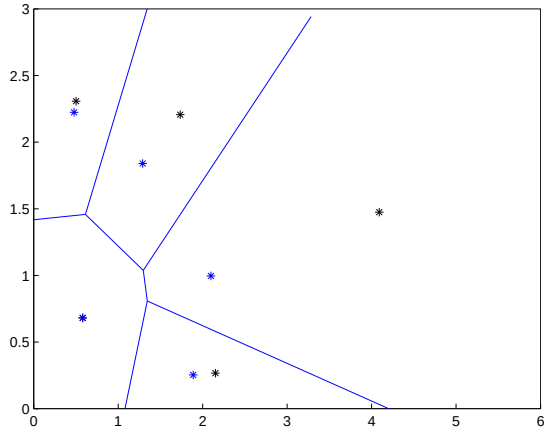


Figure 4.1: Voronoi partition generated by the robots, here in blue. The black points are the center of mass of the regions.

statement can be proved by means of (4.11)

$$\frac{\partial J_{\mathcal{V}}}{\partial p_i} = 2\left[\int_{V_i} (p_i - q)\, dq\right] = 2M_{V_i}(p_i - C_{V_i})\,.$$

The choice in (4.12) has been already used in [20].

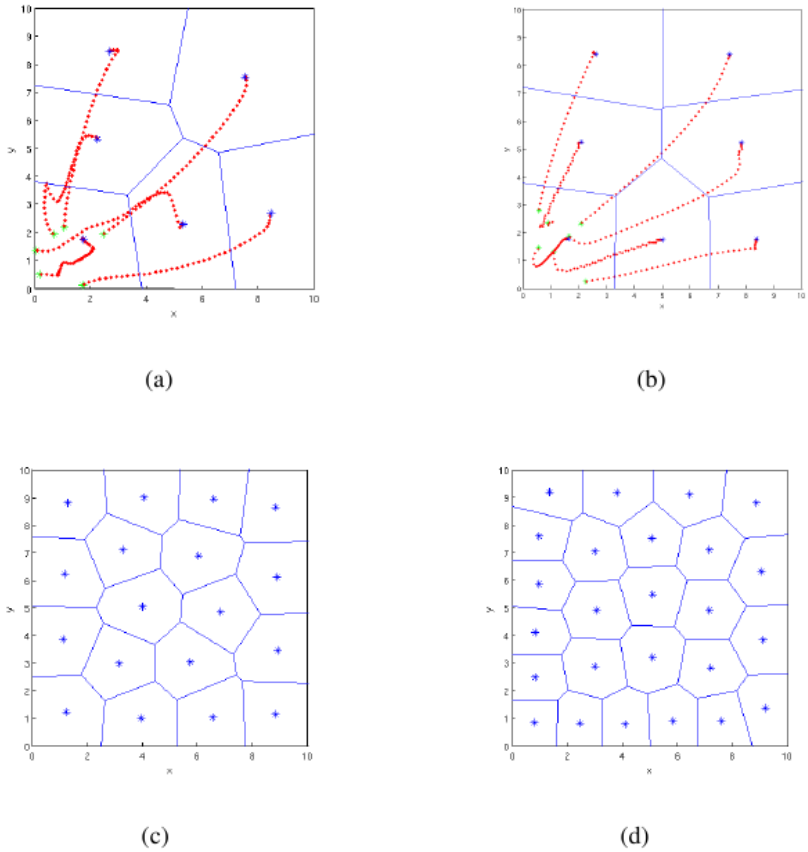
Lloyd Algorithm

Starting from an arbitrary initial robots position, one of the ways to reach the centroidal configuration is by the Lloyd algorithm [73]. The idea is the following: calculate the Voronoi partition and the respecting centers of mass, or the equivalent points (Fig. 4.1). Hence, move each robot on the center of mass of its region, or towards this one if its kinematic constraints do not allow it. Repeat this procedure for each time step until the convergence of the algorithm. We refer to [20] for more details and for the proof of the convergence. Later on, we will refer to this algorithm with LA.

Some examples of centroidal Voronoi configuration in a square convex environment with different number of robots are shown in Fig. 4.2.

4.2 Potential field approach

In this section we propose a simple modification of the classical solution discussed in Section 4.1.1 in order to cope with more complex environments. The results here presented have been presented in [95]. Hence, the aim is to study the problem of optimal placement for a team of



(a) (b)

(c) (d)

Figure 4.2: Examples of centroidal Voronoi solutions in a square area. In Fig. (a) and (b) the teams are composed respectively by six and seven robots. The initial positions are in green, in blue the final ones and in red the trajectories. In Fig. (c) and (d) the robots are respectively 18 and 25 and only the final positions are shown in blue.

mobile robots with surveillance task in a non-convex environment with obstacles. As already discussed, several papers try to tackle this more complicated problem by means of different approaches: potential field method [46], Voronoi control with geodesic distances [85], by using diffeomorphic transformations [17], or other path planning algorithm [16]. In such missions, a key factor is the *a priori* knowledge about the exact positions of the obstacles, their dimension and/or shape. We propose a very simple possible solution, based on a combination of the potential field method and the Voronoi partition, which can work even without this kind of information. Furthermore, this method has the objective to be easy to implement, with a low computational cost and it allows overcoming many of the problems of local minima, otherwise present by using only the repulsive potential field method [46]. By using an extended definition of the Voronoi regions, we also consider the case of a heterogeneous team, in which the robots have different velocity constraints. The potential field method is well known in motion planning for the obstacle avoidance problem [69], [41] and it was introduced for the first time by Khatib [54]. In [96] we proposed also another potential field based algorithm applied to a cooperative exploration problem. Regarding its applications in coverage problems, Poduri and Sukhatme adopted it in [86] to obtain a coverage of a convex region with the constraint that each robot has at least K neighbors. In [46] the region to cover is unknown and non-convex and the robots movement is due to a repulsive force generated by the other robots and by the obstacles, to obtain a dispersion of the robots.

The importance of extending the previous approach to a non-convex region is not only to describe a 2D region with obstacles but also try to develop a strategy for the 3D case, in which the non-convexity is due to land reliefs (hills, mountains, buildings and so on). The goal is to minimize the *intervention time*, i.e. the time necessary before that at least one robot reaches a given point in the space. However, because of the obstacles, it is now impossible using a cost function as in (4.4). A possible extension of (4.4) is:

$$J(\mathcal{P}) = \int_{\Omega} \min_{p_i} \tilde{d}(q, p_i)\, dq \tag{4.13}$$

where $\tilde{d}(q, p_i)$ is the distance between q and p_i, taking into account the presence of the obstacles; in other words, it is the distance that the robot p_i must cover in order to reach the point q following a feasible trajectory. First of all, this distance function is strongly environment dependent. Since in our case we assume to not know the position of the obstacles, it is neither possible expliciting the cost function in an analytical way, nor computing it at each time step. Hence, we can not use, for example, stochastic optimization methods, like SPSA [108]. In some works, for example in [46] and [86], the authors have used a potential field method to achieve the task: repulsive forces generated by all the robots and by the obstacles are consid-

ered to obtain a dispersion of the robots. The drawback is that, especially when the number of robots is small respect to the size of the environment, it is easy to find many local minima which cause suboptimal solutions. Later on, we will refer to this method with RPF (Repulsive Potential Field).

For the solution of the problem, we propose a similar approach where the robots and the obstacles (including the external boundary) produce a short range repulsive force but each robot is also attracted by the center of mass of its Voronoi region. Indeed, by adding also an attractive potential, it is possible to eliminate some local minima and the result can be improved. In other words, starting from a Voronoi-based approach, using the potential field is a simple method to only avoid obstacles if the center of mass belongs to the feasible space, or go as close as possible if it is into an obstacle. On the other hand, as in every potential filed based approach, some other local minima cannot be eliminated. Let us note that to do the Voronoi tessellation, the external boundary must be convex, but a non-convex boundary can be always approximated by a convex one with obstacles inside it.

The repulsive potential used in the model is:

$$U_{rep}(\mathbf{q}, \mathbf{q}_i) = \begin{cases} \frac{1}{2} k_{rep} \left(\frac{1}{\rho(\mathbf{q})} - \frac{1}{\rho_0} \right)^2 & , \ \rho(\mathbf{q}) \leq \rho_0 \\ 0 & , \ \rho(\mathbf{q}) > \rho_0 \end{cases} \tag{4.14}$$

where $\mathbf{q}_i$ is the position of the robot/obstacle, $\rho(\mathbf{q}) = \|\mathbf{q} - \mathbf{q}_i\|$ and ρ_0 is the range of the interaction. The artificial force induced by this potential field is $\mathbf{F}(\mathbf{q}) = -\nabla U(\mathbf{q})$:

$$\mathbf{F}_{rep}(\mathbf{q}, \mathbf{q}_i) = \begin{cases} k_{rep} \left(\frac{1}{\rho(\mathbf{q})} - \frac{1}{\rho_0} \right) \frac{\mathbf{q} - \mathbf{q}_i}{\rho^3(\mathbf{q})} & , \ \rho(\mathbf{q}) \leq \rho_0 \\ 0 & , \ \rho(\mathbf{q}) > \rho_0 \end{cases} \tag{4.15}$$

Then, each robot feels a total repulsive force equals to:

$$\mathbf{F}_{rep}(\mathbf{q}) = \sum_{i=1}^{N} \mathbf{F}_{rep}(\mathbf{q}, \mathbf{q}_i) \tag{4.16}$$

where the sum is over the other $N - 1$ robots and the closest obstacle.

The attractive potential used is:

$$U_{att}(\mathbf{q}) = \frac{1}{4} k_{att} \, \rho_{goal}^4 \tag{4.17}$$

and the corresponding force

$$\mathbf{F}_{att}(\mathbf{q}) = k_{att}(\mathbf{q}_{goal} - \mathbf{q})\,\rho_{goal}^2 \tag{4.18}$$

where $\rho_{goal} = \|\mathbf{q} - \mathbf{q}_{goal}\|$ and $\mathbf{q}_{goal}$ is, in our case, the center of mass. Added to these forces, we also consider a viscous term, νv, in order to have more regular trajectories. The equation of motion is:

$$\mathbf{F}_{tot} = \mathbf{F}_{rep} + \mathbf{F}_{att} = m\,\ddot{\mathbf{q}} - \nu\,\dot{\mathbf{q}} \tag{4.19}$$

where m is the virtual robot mass which, without any loss of generality, we assume unitary. We will refer to our approach with RAPF (Repulsive and Attractive Potential Field).

4.2.1 Heterogeneous team

Since we try to achieve a static deployment that optimizes the intervention time problem, a natural extension of the proposed algorithm is to take into account the case in which the team includes robots with different velocity capabilities. A possible way to do it is changing the definition of the Voronoi region in the following way. Let p_i be the position of the $i-th$ robot and v_i its maximum velocity, we define the new Voronoi regions

$$V_i = \{q \in \Omega \mid \quad \frac{1}{v_i}\|q - p_i\| \leq \frac{1}{v_j}\|q - p_j\| \ \forall j \neq i\}\,, \tag{4.20}$$

which is the well-known expression of the weighted Voronoi partition (see [82]). To find the analytical expression of the boundaries for these regions, without any loss of generality, we put the robot i in the origin of the frame system and the other one in $p = (p_x, p_y)$. In the classical homogeneous case, the edge shared is given by

$$x^2 + y^2 = (x - p_x)^2 + (y - p_y)^2 \Rightarrow p_x^2 + p_y^2 - 2(xp_x + yp_y) = 0 \tag{4.21}$$

which is the equation of a straight line. Indeed, the Voronoi regions are polygons. If we consider different velocities we have:

$$\begin{aligned} &k(x^2 + y^2) = (x - p_x)^2 + (y - p_y)^2 \\ &\Rightarrow x^2(k-1) + y^2(k-1) + 2(xp_x + yp_y) - p_x^2 - p_y^2 = 0 \end{aligned} \tag{4.22}$$

where $k = v_j^2/v_i^2$, which defines an arc of circle. This is more evident if we rewrite the equation (4.22) in the following way:

$$\left(x + \frac{p_x}{k-1}\right)^2 + \left(y + \frac{p_y}{k-1}\right)^2 - k\left[\left(\frac{p_x}{k-1}\right)^2 + \left(\frac{p_y}{k-1}\right)^2\right] = 0 \tag{4.23}$$

that is the equation of a circle of center

$$c = \left(-\frac{p_x}{k-1}, -\frac{p_y}{k-1}\right) \tag{4.24}$$

and radius

$$r = \sqrt{k\left[\left(\frac{p_x}{k-1}\right)^2 + \left(\frac{p_y}{k-1}\right)^2\right]} = \sqrt{k}\, d_c = \frac{v_j}{v_i}\, d_c \tag{4.25}$$

where d_c is the distance of the center from the robot i. We can interpret $\sqrt{k}\, d_c$ like the distance traveled by the robot j in the time that i reaches the center c. It is possible to write r also in terms of the distance between the two robots, d_r:

$$r = \frac{\sqrt{k}}{|k-1|}\, d_r = \frac{v_i\, v_j}{|v_i^2 - v_j^2|}\, d_r\,. \tag{4.26}$$

Let us note that if

$$v_i \ll v_j \rightarrow r = \frac{v_i}{v_j}\, d_r\,. \tag{4.27}$$

We finally remark that when $v_i \rightarrow v_j$ we obtain the standard Voronoi tessellation. Indeed,

$$\lim_{v_i \rightarrow v_j} r = \infty\,, \tag{4.28}$$

which means that we have an arc of circle with infinite radius, i.e., a straight line.

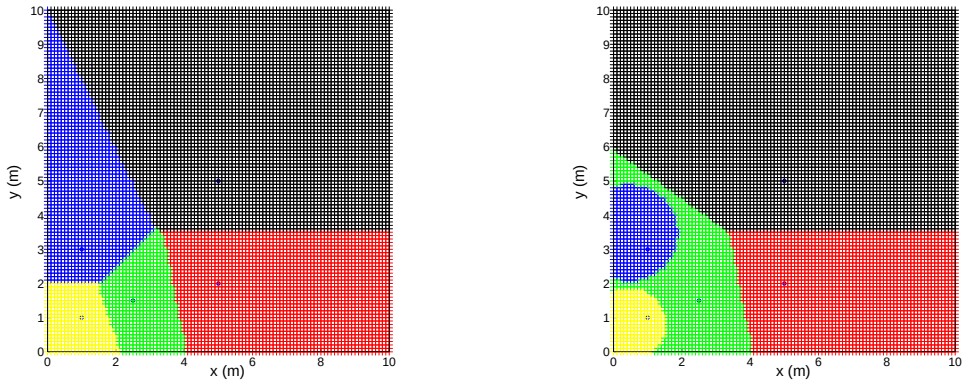
An example of the difference between the classical definition of the Voronoi regions for a homogeneous team and the weighted one is shown in Fig. 4.3.

To take into account the different speeds of the robots in the cost function, we can extend the expression (4.4) as follows:

$$J(\mathcal{P}) = \int_\Omega \min_i \frac{1}{v_i}\, \tilde{d}(q, p_i)\, dq\,. \tag{4.29}$$

Finally, we note that with a simple repulsive potential field approach it is not possible to take into account information about different robots velocities.

To better see the difference between homogeneous and heterogeneous teams, we apply



(a) (b)

Figure 4.3: The difference between the homogeneous case and the heterogeneous case. In (a) all the robots have the same velocities, in (b) there are three faster robots and two slower. It is evident the different shape of the regions.

the LA algorithm for the coverage of a convex environment using the different definitions for the Voronoi regions. The simulations are shown in Fig. 4.4, where it is possible to see how trajectories and final configurations change with varying robots speeds.

4.2.2 Simulation results

We propose some numerical simulations by using teams of different number of robots and in different environments. To have a quantitative result, we have computed the cost functions (4.13) and (4.29) by a discretization of the space to obtain the distance between each site of the lattice and the closest sensor. We assume that the robots know exactly their position in a common frame of reference. In this simulation we have considered only the center of mass of the Voronoi regions as the optimal point to reach, i.e. making the choice (4.12), because it is the most suitable for our intent. Furthermore, we compare the proposed algorithm with the RPF to show that in a generic environment it is able to improve the result.

A first example of the algorithm is shown in Fig. 4.5. In this case the team is composed by 15 robots, which have to cover a generic 2D non-convex environment. We can see from the robots trajectories (Fig. 4.5 (a)) that there is a spreading out of the team from the initial positions to a well-distributed final configuration.

Our intent is also to show that our algorithm outperforms the RPF method. In Fig. 4.7 there are the trajectories of 5 mobile robots. They are generated, starting from the same initial robots configuration, by using the RPF method and the RAPF (for simplicity we consider here

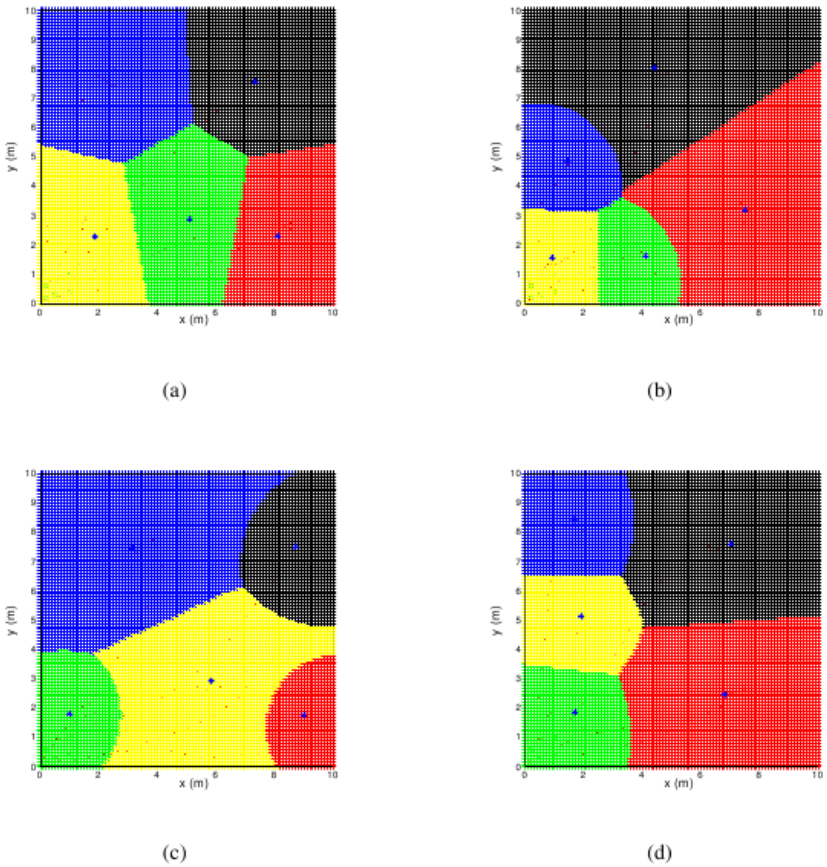


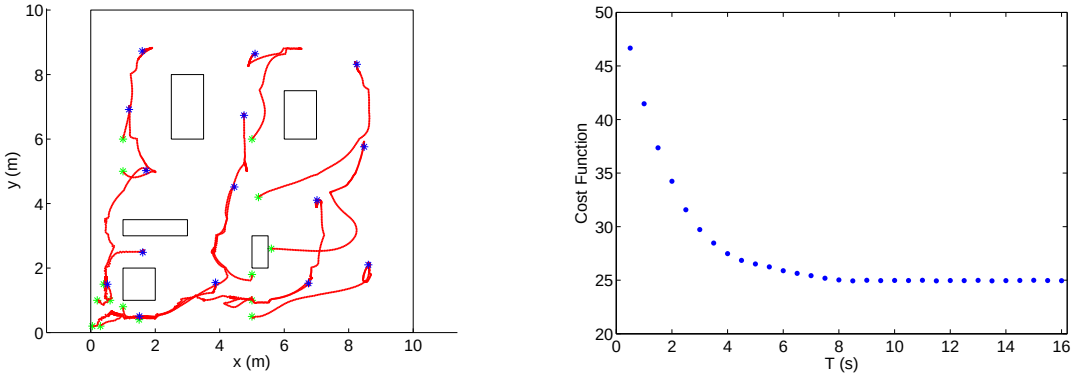
Figure 4.4: Differences between the homogeneous case and the heterogeneous case. In (a) a homogeneous team ($k = 1$), the solution is the centroidal partition; in (b) and (c) a heterogeneous one with two faster robots and three slower ($\sqrt{k} = 0.5$); in (d) different k-factor ($\sqrt{k} = 0.6$).

a homogeneous team). It is possible to see from the figure that the robots are more spread out by means of our algorithm. Indeed, this is a typical example where the robots are not able to overcome a barrier of obstacles, and the environment is not well covered. To have a significant feedback of the improvement, we have generated 100 different environments, with the number of obstacles varying from 1 to 6. The obstacles are always of fixed size and their positions into the space are randomly generated. The boundary of the region to cover is a square of fixed size. Obviously, some constraints for the positioning of the obstacles have been considered: for example they can not form barriers which forbid at all the motion of the robots from their initial position. For each environment, we have calculated the final value of the cost functions obtained by the RAPF and the RPF. In Fig. 4.6 the result is presented. It can be seen that the cost function for the RAPF is always lower. Furthermore, the values of the RPF are more scattered around a mean value compared to that obtained with the RAPF. The reason is that the best choice of the potential parameters is environment dependent if we do not consider the Voronoi attraction. On the other hand, by using the RAPF, it is almost independent: this is a fundamental aspect if we do not have any a priori information about the obstacles. This result allows us to state that the proposed algorithm is an improvement of the already existing method.

Finally, we want to emphasize the importance of considering the weighted definition of the Voronoi regions for the heterogeneous case. To prove it, we compare, for a heterogeneous team, the cost function obtained considering the weighted Voronoi partition with that one obtained by the same algorithm, but using the usual definition of the Voronoi regions. In other words, we compare the cost functions relative to the simulation shown in Fig. 4.7 (a), but consider the team as heterogeneous to calculate the cost function. Of course, the expression of the cost function taken into account is the one in (4.29). The result in Fig. 4.8 shows that with the new definition of the Voronoi regions the cost function is lower.

4.3 Visibility Coverage

We consider now the first objective described at the beginning of this chapter: the visibility problem. This problem is more important for our final propose of surveillance in a real and complex environment. Indeed, an optimal solution obtained starting from the objective function (4.4) might allow the team to see a very low part of the environment. On the other hand, due to the non-convex nature of the problem, an analytical solution for this problem cannot be obtained. To approach this problem, we propose a new solution that is based on the previously described Cognitive-based Adaptive Optimization (CAO) algorithm. We recall



(a) (b)

Figure 4.5: Example of coverage of an environment with obstacles by using a team of 15 mobile robots. In (a) are shown the trajectories: the initial positions are in green, the final ones in blue. In (b) the cost function.

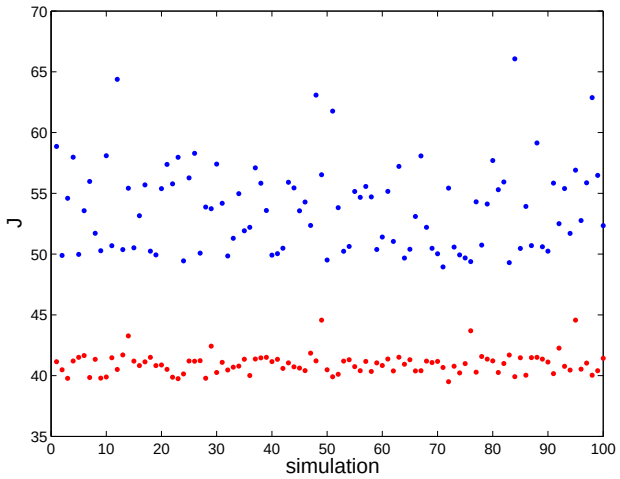
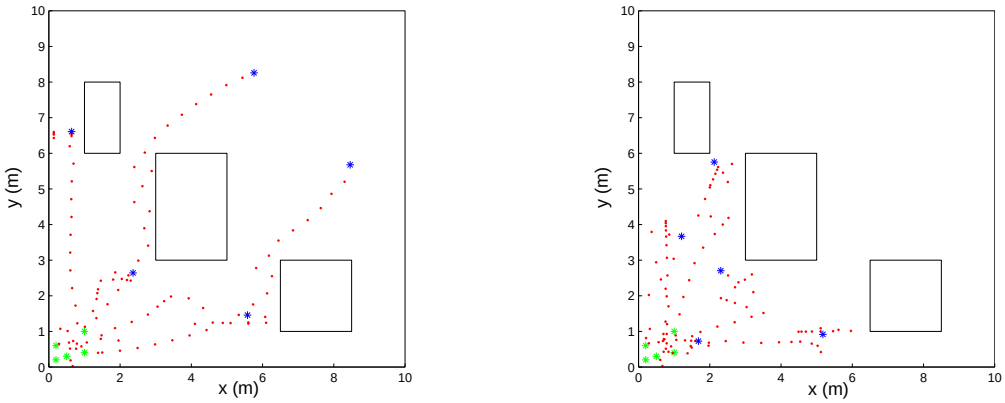


Figure 4.6: Final cost functions obtained by the RAPF method, in red, and by the RPF method, in blue, for each environment randomly created.



(a) (b)

Figure 4.7: Simulation with an environment with obstacles. In (a) the coverage is obtained with a homogeneous team by using the RAPF algorithm. In (b) the coverage obtained by using the RPF algorithm.

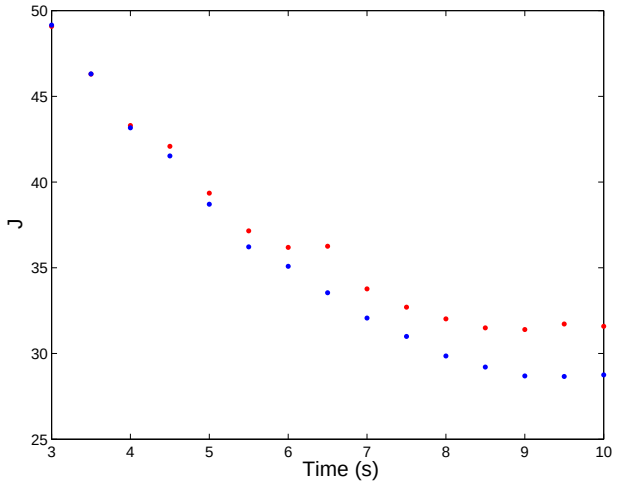


Figure 4.8: Cost functions for a heterogeneous team in the environment shown in Fig. 4.7 obtained by the RAPF. In blue the cost function obtained by using the weighted definition of the Voronoi regions, in red by using the usual definition.

here that the main advantage of CAO as compared to standard optimization tools is that it does not require that the objective function to be optimized is explicitly known; CAO instead requires that at each-time instant a value (measurement) of this objective function is available. Then, if it is possible to define an objective function which is available for measurement for every given team configuration, the CAO methodology will be directly applicable to the problem of surveillance coverage treated in this thesis. Rigorous arguments establish that, despite the fact that the terrain's morphology is unknown, the CAO methodology shares the same convergence characteristics as those of constrained gradient-descent algorithms, which require perfect knowledge of the terrain's morphology to optimize the surveillance coverage, subject to the constraints the robot team has to satisfy. As a result, CAO navigates the robots to an arrangement that locally optimizes the surveillance coverage criterion while satisfying physically-imposed constraints such as that the robots should not leave a prespecified area or they should not hit the obstacles.

4.3.1 Centralized Algorithm

Mathematically, we can define the problem in the following way. Let us consider a planar non-convex environment and let Ω be the region accessible by the robots. Let $\mathcal{P} = \{x_k^{(i)}\}_{i=1}^{M}$ denote the position of the M robots at the time step k and $C = \{r_i\}_{i=1}^{M}$ the maximum distances of monitoring for the $i-th$ robot. In our approach, we consider the monitoring of a point $q \in \Omega$ a binary function

$$f(q, \mathcal{P}; C) = \begin{cases} 1 & \text{if } q \text{ is monitored} \\ 0 & \text{otherwise} \end{cases} \tag{4.30}$$

Let us assume that a robot can monitor the points which satisfy both the following conditions:

- are connected by a line-of-sight with it;
- are at a distance smaller than a given threshold value.

In Fig. 4.9 it is shown an example of the monitored area given the positions of two robots and a maximum monitoring distance of $5m$ for both the robots.

Thus, we can define the cost function $\mathcal{J}$ as follows:

$$\mathcal{J}(\mathcal{P}; C) = \frac{1}{V} \int_{\Omega} f(q, \mathcal{P}; C)\, dq \tag{4.31}$$

where $V = \int_{\Omega} dq$. Obviously, this is only an implicit expression of the cost function and it is impossible to get an explicit form because of the dependency on the particular environment.

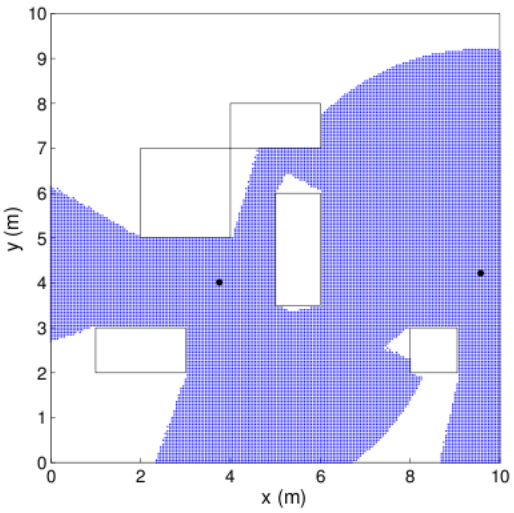


Figure 4.9: Example of area monitored by two robots equipped with omnidirectional visual sensor in a non-convex environment. The dots correspond to the robots' locations, the rectangles denote the obstacles.

However, as explained in chapter 3, we just need the numerical value of the cost function for each time step and not its explicit expression. This is the key advantage of CAO which does not require an a priori knowledge of the environment.

Having defined the optimization problem, a fundamental issue for a good behavior of the CAO algorithm is an appropriate choice of the form of the regressor vector ϕ, introduced in equation (3.11). As mentioned in chapter 3, several different choices for its explicit expression are admissible. However, in all different tests for the particular application treated in this thesis, it was found that it suffices to choose the regressor vector as follows:

1. choose the size of the function approximator L to be an odd number;

2. select the first term of the regressor vector ϕ to be the constant term;

3. select randomly the next $(L-1)/2$ terms of ϕ to be any 2nd-order terms of the form $x_a^{(i)} \cdot x_b^{(j)}$ [with $a, b \in \{1, \ldots, \dim(x^{(i)})\}, i, j \in \{1, \ldots, M\}$ randomly-selected positive integers];

4. select the last $(L-1)/2$ terms of ϕ to be any 3rd-order terms of the form $x_a^{(i)} \cdot x_b^{(k)} \cdot x_c^{(j)}$ [with $a, b, c \in \{1, \ldots, \dim(x^{(i)})\}, i, k, j \in \{1, \ldots, M\}$ randomly-selected positive integers].

Once the regressor vector ϕ has been set and once the values of the cost function (4.31) are available for measurement at each time step, it is possible to find at each time step the

vector of parameter estimates θ_k and thus the approximation of the cost function $\hat{J}_k$. The other important choice in order to assure the convergence of the algorithm is the expression of the sequence α_k, defined in equation (3.19). A typical choice for such a sequence is given by

$$\alpha_k = \frac{\alpha}{(k+1)^\eta}\,, \tag{4.32}$$

where α is a positive user-defined constant and $\eta \in (0, 0.5)$.

4.3.2 Distributed Algorithm

In every multi-robot systems, a distributed approach is desirable for several fundamental reasons. The most important are failure of the central station and limited communication capabilities. In a very common scenario each robot has no global knowledge about the surrounding environment or about the group as a whole. Hence, the global behavior of the team can be seen as the sum of the local actions taken by its members, which sense their immediate environment, communicate with their neighbors, process the information gathered and move according to it. Even if the optimization function (4.31), which we want to maximize, depends on all the state vector's variables, it is reasonable to think that the coverage problem could be expressed in a decentralized way. Indeed, the aim of each robot is to deploy itself in order to minimize the overlapping of its field of view with the obstacles in the environment and with the fields of view of the other robots. In other words, for each robot the problem is like maximizing the surface monitored while it is moving in an environment with both static obstacles and dynamic obstacles, which are the fields of view of the other robots (see Fig. 4.10). In practice, we can write the new optimization function for the robot i as follows:

$$J_i(\mathcal{P}) = V_i - \sum_{j \neq i} V_i \bigcap V_j\,, \tag{4.33}$$

where V_j is the total area monitored by the $j-th$ robot.

In other words, instead of a single cost function, the problem is now characterized by N cost functions, being N the number of robots. The CAO approach is adopted to independently maximize each optimization function. In particular, each function will be characterized by a different approximator. Then, the CAO approach will perform the optimization by only perturbing the position of the respecting robot.

It is important to note that at each time step this new single-robot optimization function depends only on the position of the robots which have an overlapping field of view, which may be identified as the neighbors. This means that, if we define with R_i the maximum monitoring

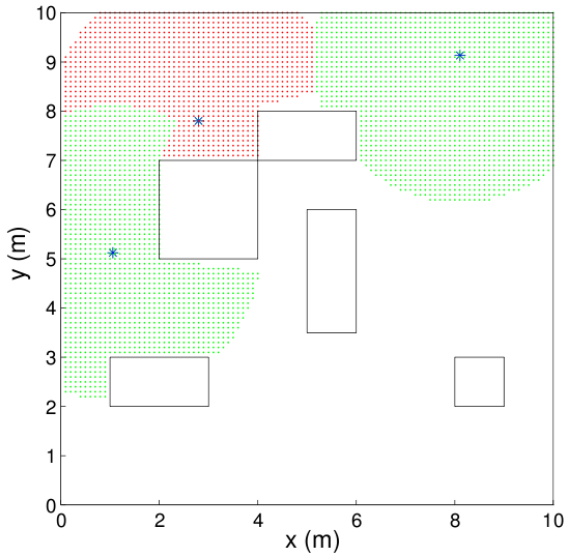


Figure 4.10: Example of area monitored by three robots equipped with omnidirectional visual sensor in a non-convex environment (the blue dots correspond to the robots' locations, and the rectangles denote the obstacles). The field of view in red is the effective monitored area (optimization function) of the respective robot.

distance of the robot i, at a given time step its objective function depends on the position of a robot j if and only of

$$\|p_i - p_j\| < R_i + R_j$$

However, to obtain the approximator, the CAO algorithm uses a set of past stored data (see eq. 3.12). It is thus necessary to consider also robots that have been neighbors in these past time steps. On the other hand, information about robots which, during this time interval, have always been more distant is completely useless and their positions are not taken into account for the construction of the approximator.

For this reason, the proposed distributed method can also take into account communication constraints. Indeed, each robot has to know only the real positions of the other robots when they are neighbors. For robots which are not neighbors, i.e. for the robots whose monitored area does not overlap with the area of the considered robot, knowing the actual positions is unnecessary. Practically, if some of these unknown positions must be used for the approximator, the considered robot generates for them fictitious regular trajectories. These false positions do not influence the construction of the approximator because it does not depend on them. Then, if one of these robots returns to being a neighbor it can transmit its real past positions and the approximators is calculated with all the right values. Therefore, every robot has to communicate only with its neighbors and even if there is a failure in this communication the algorithm can continue to work.

4.4 Simulation results

To evaluate the efficiency of the proposed algorithm, several simulations with varying number of robots and different monitoring constraints, have been performed in a variety of environments. We show the results of the simulations comparing the performance of the centralized approach (i.e. the one which maximizes (4.31) by simultaneously perturbing the whole state vector) and the distributed approach introduced in section 4.3.2. The teams are considered to be homogeneous since the maximum distance of monitoring for each robot is the same, although it is not the same in all simulated scenarios. This assumption has been made for simplification purposes and easier comprehension of the results. Different robots monitoring capabilities within the team do not affect in any way the efficiency of the algorithm.

4.4.1 Visibility

We present now the results for the visibility problem in a non-convex environment. In the first simulations we assume the robots are equipped with an omnidirectional camera. Then, some simulations with a limited field of view of the robots are also provided. No assumption on the topology of the environment has been taken, and the obstacles in the simulations here presented are always polygonal only for simplicity's sake.

Limited-range omnidirectional monitoring

As a first test, we consider the trivial case of a convex environment. Indeed, this case has been extensively considered in the literature and the solution is known [20]. In Fig. 4.11, we show that the proposed method is able to reproduce such solution, which corresponds to the centroidal Voronoi partition where the robots' positions are the generators of the partition. This result is an important test for our method, although the main objective of our work is to study the coverage problem in a more realistic scenario. We remark that for the simply visual-based problem, with no restriction on the maximum monitoring distance, for the convex case every different robots' placements are completely equivalent.

In the second simulation presented in Fig. 4.12, the team is composed by three robots with a maximum monitoring distance $r = 5m$. The cost function, in Fig. 4.12(b), indicates that the algorithm is able to provide a very good solution. The efficiency of the proposed solution can also be evaluated by observing the robot trajectories in Fig. 4.12(a). The robots move in order to eliminate all the shadow regions generated by the obstacles and to minimize the overlapping zones monitored by more than one sensor.

Fig. 4.13(a) shows a typical scenario in which four robots have to cover a nonconvex

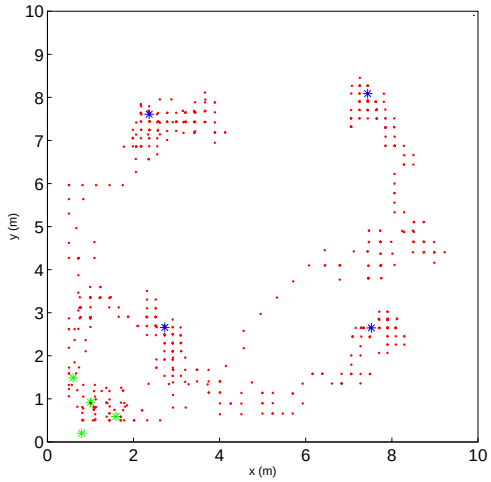
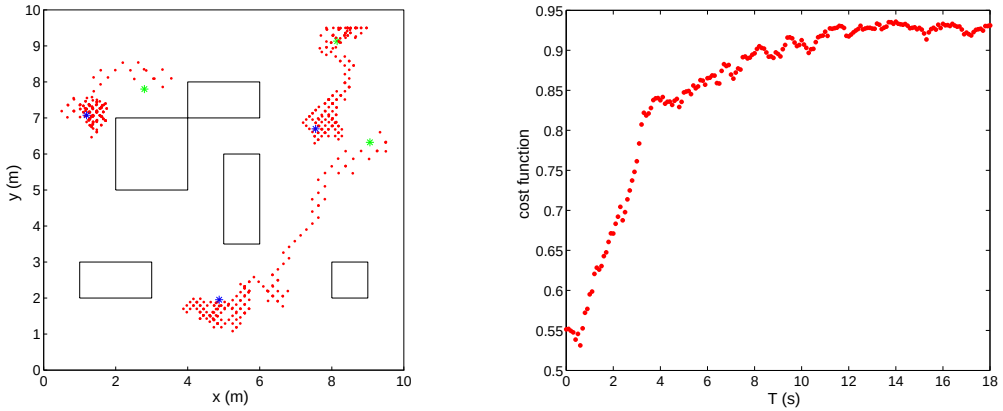
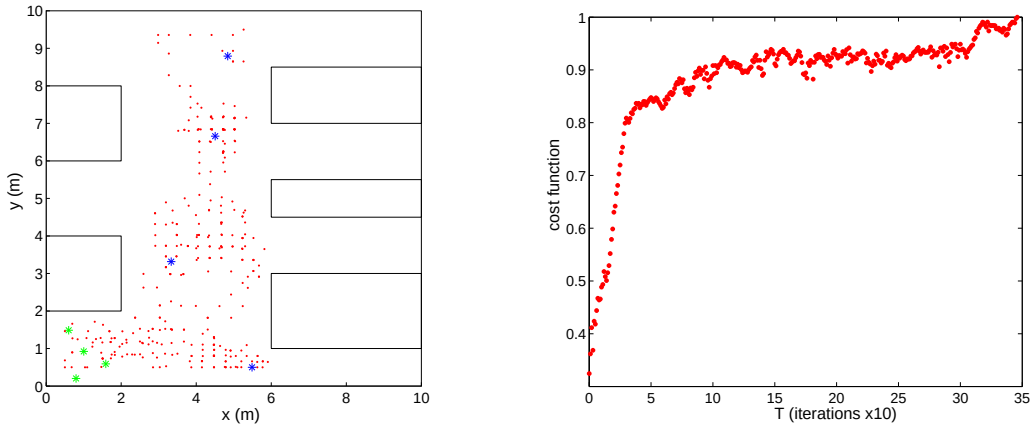


Figure 4.11: Four robots with a maximum monitoring distance $r = 3.5m$ in a convex environment. The green points show the initial positions of the robots, the final ones are in blue, in red the trajectories. The solution reproduces the centroidal Voronoi solution, where the robots' positions are the generator of the partition.



(a) (b)

Figure 4.12: Three robots with a maximum monitoring distance $r = 5m$.



(a) (b)

Figure 4.13: Centralized approach. Four robots with a maximum monitoring distance $r = 6m$. In Fig. (a) it is shown the robots' motion: the green points show the initial positions of the robots, the final ones are in blue, in red the trajectories. In Fig. (b) the cost function $\mathcal{J}(\mathcal{P})$.

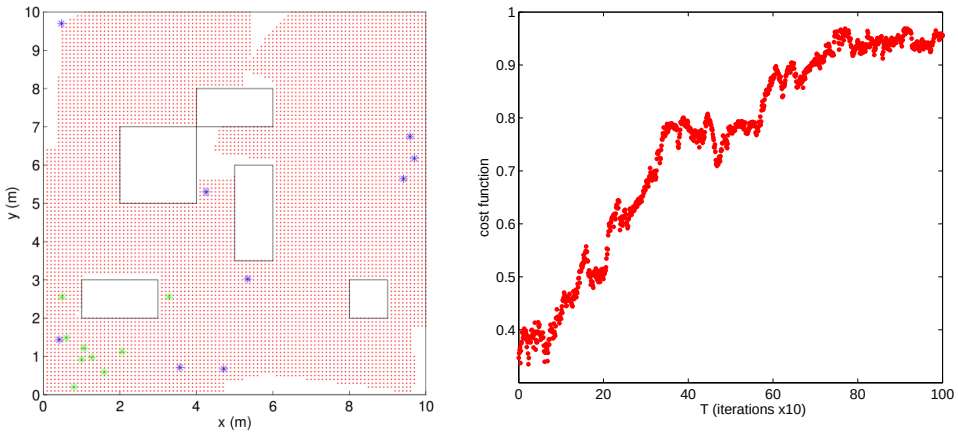
environment. The behaviour of the cost function during the task is shown in Fig. 4.13(b). It is possible to see that the final deployment is such that the team is able to monitor the whole environment.

Limited-range anisotropic monitoring

Here we present results for the more complicated case in which each robot has a limited visibility angle. Two factors make this case more complicated than the previous: first of all each robot has now one degree of freedom more, which is its orientation; moreover, the objective function becomes more irregular and consequently more complicated to optimize. In order to have a simple comparison with the omnidirectional vision case, we have used the same environments of the previous simulations. Since now it is important not only the robots' positions but also their orientation, for clarity's sake we have shown explicitly in the figures the monitored area for each robot (Fig. 4.14(a), 4.15(a)).

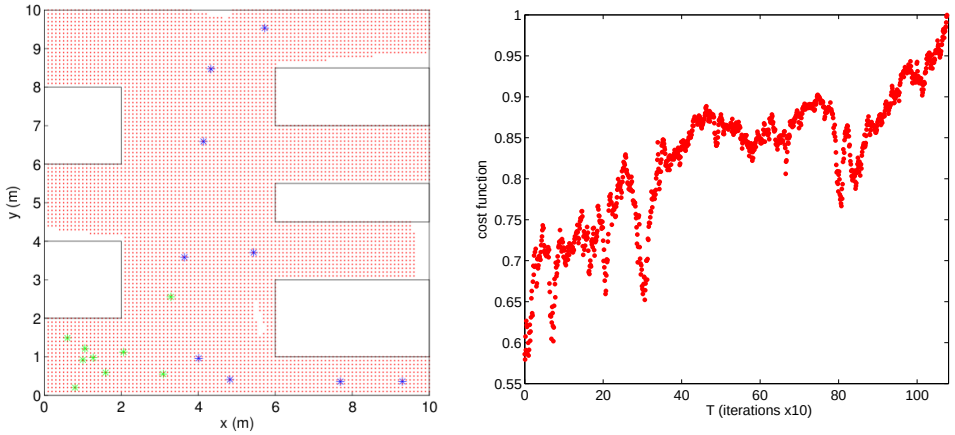
By observing the cost function (Fig. 4.14(b), 4.15(b)) it is evident that in this case it is less smooth. Because of this the proposed method may need more iterations to converge.

Another consequence of this fact is that the number of local maxima increases significantly with respect to the omnidirectional case, and sometimes they can force the system to local solutions very far from the optimal configuration. An example where it can be easily seen this phenomenon is shown in Fig. 4.16. As always in presence of difficult local optima problems,



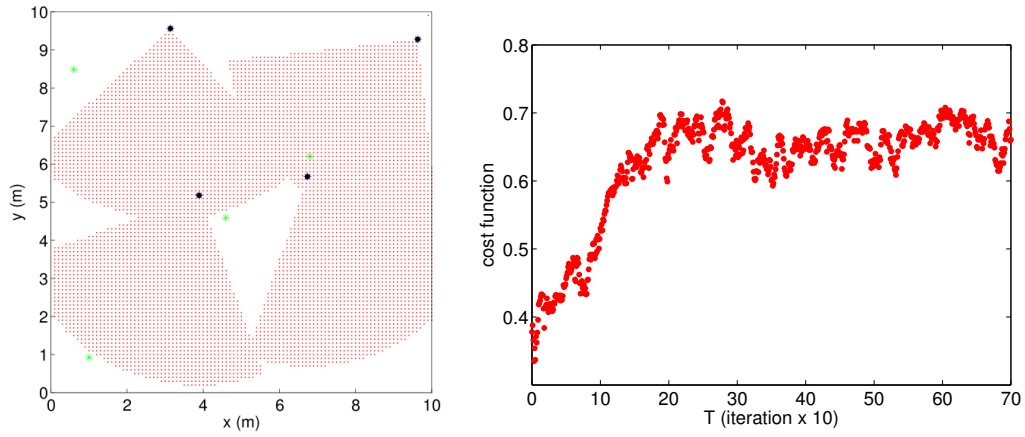
(a) (b)

Figure 4.14: Nine robots with a maximum monitoring distance $r = 5m$ and 120 degree of monitoring angle.



(a) (b)

Figure 4.15: Nine robots with a maximum monitoring distance $r = 6m$ and 120 degree of monitoring angle.



(a) (b)

Figure 4.16: Four robots in a convex square area. It is clear that, due to a local minimum, the algorithm cannot find the best solution.

the final positions strongly depend on the initial ones.

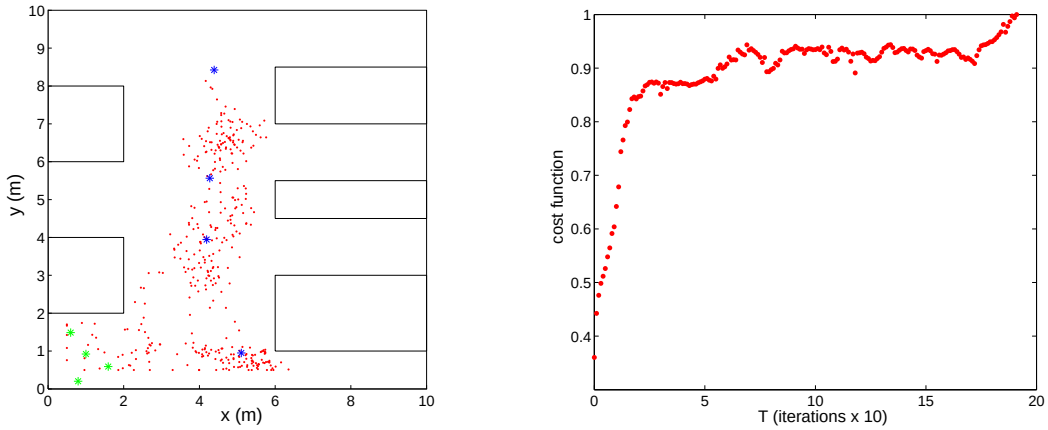
Distributed algorithm

After the results for the standard centralized version of the proposed algorithm, we show now that it is possible to obtain the same achieved results by using the distributed solution. In Fig. 4.17(a) there is the result obtained starting from the same configuration than for the centralized case previously presented (Fig. 4.13(a)). It is worth to note that, as it can be seen in Fig. 4.17(b), the time of convergence is even lower than that one obtained by employing the centralized approach. This unexpected behavior deserves a further explanation. Even though a thorough analysis of these results is still to do, our interpretation is based on the fact that with the same amount of calculation, in the distributed implementation a larger portion of configuration space is explored and each robot can straightly optimize its own position.

In the last proposed simulation (Fig. 4.18) the team is composed by six robots with a monitoring radius of $r = 3m$ and the environment has different obstacles. Also in this case the team is able to find a final configuration which allows monitoring almost all the free space and the distributed algorithm is faster to converge (see Fig. 4.18(b)).

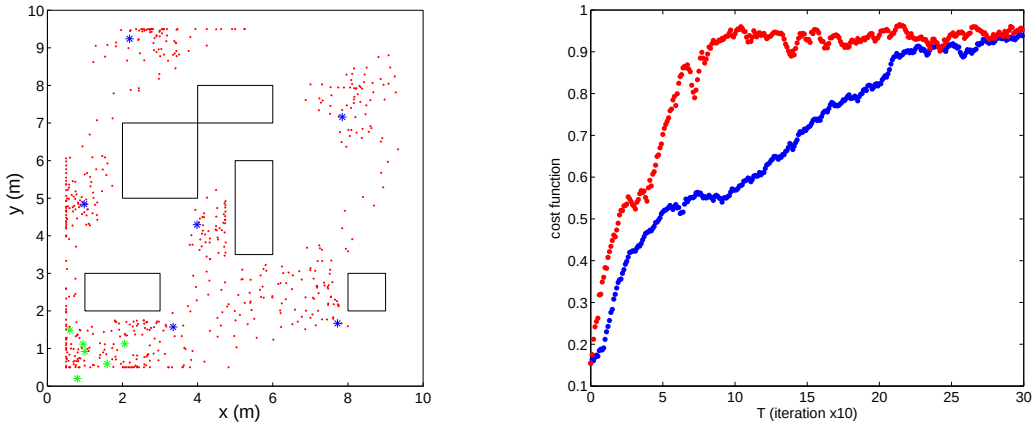
4.5 Conclusions

This chapter presented the formulation of the cooperative surveillance coverage for the case of a 2D region. After beginning with the description of the well known local solution of the



(a) (b)

Figure 4.17: Distributed approach. Four robots with a maximum monitoring distance $r = 6m$.



(a) (b)

Figure 4.18: Six robots with a maximum monitoring distance $r = 3m$. In Fig. (b) the cost functions obtained by using the two versions of the algorithm: in red the distributed method, in blue the centralized one.

Voronoi coverage control, as a first contribution we presented a simple possible extension, based on the artificial potential field method, for the case of a non-convex region with unknown obstacles. Then, we introduced the visibility problem in a complex 2D environment. This topic is more important for the purpose of this thesis and we explained why the proposed stochastic optimization algorithm is suitable for this problem. The 2D case is simpler than the 3D, but it has several advantages for a first application and evaluation of the proposed method. The most important is to try to better understand the problem and the behavior of the algorithm in a case where the optimal solution is known or, at least, easier to figure out. We consider the results here obtained as fully satisfying and they are the starting point to go on with this work for a 3D application, presented in the next chapter.

Chapter 5

Optimal Surveillance Coverage - 3D

5.1 Problem Definition

In the previous chapter we introduced the concept of cooperative surveillance coverage and we extensively described the case of using the CAO approach for maximizing the monitored area in a given region by using a team of mobile robots in the 2D plane, without any assumption on the topology of the environment. In this section we extend our approach to the more interesting case of robots living in a 3D environment and having fixed orientation.

Let us consider a team of M flying robots that is deployed to monitor an unknown terrain $\mathcal{T}$. Let z denote the unknown height of the terrain at the point (x, y) and assume for simplicity that the terrain $\mathcal{T}$ is rectangular along the (x, y)-axes, i.e., $x_{min} \leq x \leq x_{max}, y_{min} \leq y \leq y_{max}$. Let $\mathcal{P} = \{x^{(i)}\}_{i=1}^{M}$ denote the configuration of the robot team, where $x^{(i)}$ denotes the position of the i-th robot. We will say that a point $q = (x, y, z)$ of the terrain is visible if there exists at least one robot so that

- the robot and the point q are connected by a line-of-sight;
- the robot and the point q are at a distance smaller than a given threshold value (defined as the maximum distance the robot's sensor can "see").

Given a particular team configuration $\mathcal{P}$, we denote with $\mathcal{V}$ the visible area of the terrain, i.e., $\mathcal{V}$ consists of all points $q \in \mathcal{T}$ that are visible from the robots. We will assume that the robots are equipped with visual sensors together with inertial sensors and/or range sensors; in other words, for each visible point we will assume that the team is able to estimate the terrain's height at this point. A possible way to deploy a robot team satisfying the above, is by using the down-looking-camera-equipped flying robots of [11, 119] which employ the monocular SLAM algorithm of [58].

5.2 Algorithm Implementation

An efficient trajectory generation algorithm for optimal coverage must make sure that the physical constraints are also met throughout the whole multi-robot coverage application. Such physical constraints include, but are not limited to, the following ones:

- 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 $[z + d, z_{max}]$ along the z-axis, where d denotes the minimum safety distance (along the z-axis) the robots' should be from the terrain and z_{max} denotes the maximum allowable height for the robots.
- The robots do not come closer to the other ones than a minimum allowable safety distance d_r.

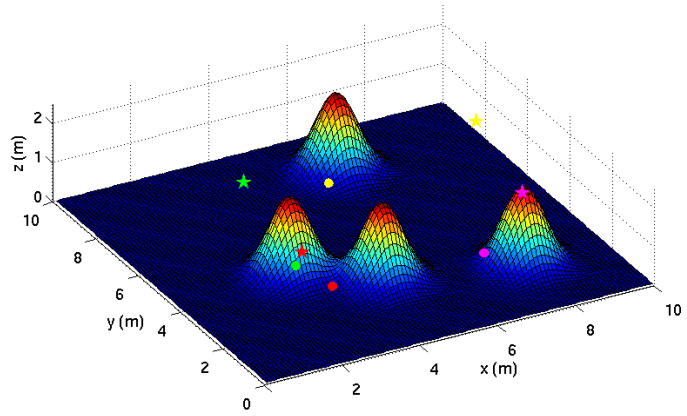
It is not difficult to see that all the above constraints can be easily expressed in the form (3.3) and thus can be handled by the CAO algorithm.

5.3 Distributed Algorithm

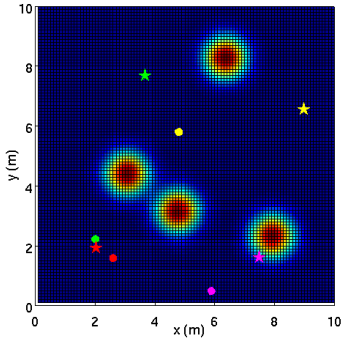
We present here the straight extension for a 3D region of the distributed version of the coverage algorithm presented in section 4.3.2. The strategy to reduce the problem to several single-robot objective functions to separately optimize is unchanged and the coverage criterion is alway defined as the total percentage of surface seen by the team. For the 2D case we could see the problem as if each robot is moving in a region with both static and dynamic obstacles, which are the field of view of the other robots. In the same way, now we can reformulate this distributed version as follows: each robot tries to optimize the area he is able to monitor on a surface which includes moving regions without interest, which are the portions that the other robots of the team can monitor. This allows the robots to maximize the total covered area avoiding unnecessary zones of overlapping. The same strategy adopted in section 4.3.2 to incorporate limited communication capabilities is used also in this case.

5.3.1 Simulation results

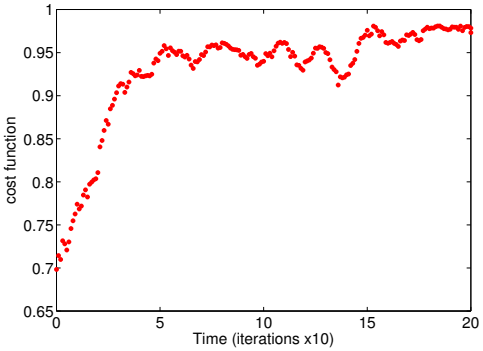
We show here the results of the simulations, comparing the performance of the centralized approach (i.e. the one which maximizes (4.31) by simultaneously perturbing the whole state



(a)



(b)



(c)

Figure 5.1: Distributed approach. Four robots with a maximum monitoring distance $r = 4m$. Fig. (a) shows the initial and final positions, as circles and stars respectively, in the 3D environment. Different colors correspond to different robots. In Fig. (b) the same result is projected on the xy plane. The behavior of the objective function is in Fig. (c).

vector) and the distributed approach introduced in section 4.3.2. The teams are considered to be homogeneous since the maximum distance of monitoring for each robot is the same, although it is not the same in all simulated scenarios. This assumption has been made for simplification purposes and easier comprehension of the results.

Fig. 5.1 (a) and (b) presents a typical scenario in which four robots have to cover a 3D region. In the figure are shown the initial, random, positions and the final configuration obtained by the CAO algorithm. The environment has been constructed as a sum of four Gaussians of same height. The behaviour of the cost function during the task is shown in Fig. 5.1(c). It is possible to see that the final deployment is such that the team is able to monitor the whole

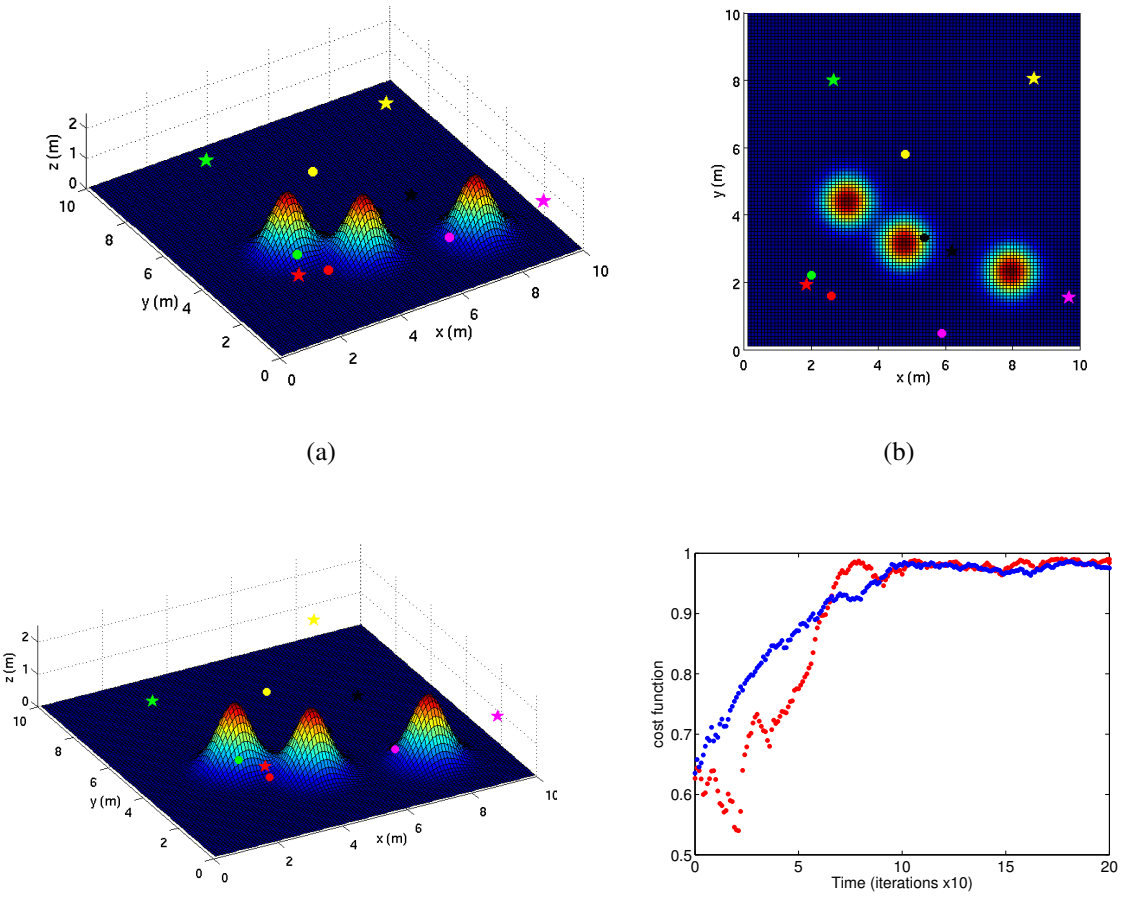


Figure 5.2: Centralized versus distributed approach. Five robots with a maximum monitoring distance $r = 3m$. In Fig. (a) and (b) is shown the result obtained using the distributed algorithm. In (c) the same simulation with the same parameters has been carried out using the centralized version. Fig. (d) compares the objective functions: in red the distributed algorithm and in blue the centralized one.

environment.

Then, we want to prove that it is possible to obtain the same results achieved with the centralized algorithm by using the distributed solution. In Fig. 5.2(a) and (b) it is presented the result obtained with a team of five robots with a maximum monitoring distance of $5m$ in a region where three Gaussians are present. In Fig. 5.2(c), the same simulation with the same starting positions has been done by employing the centralized version of the CAO algorithm. First of all we can straight notice that the final positions are very similar in the two cases. Then, by comparing the behavior of the objective functions, as shown in Fig. 5.2(d), we have the demonstration that the two methods converge at the same value, i.e. almost the entire area is covered. Contrary to the results obtained in section 4.3.2, where the distributed algorithm was faster to converge, in this case the algorithms have almost the same time of convergence. Anyway, these results are completely satisfactory since what we expected was a slightly worse performance, which would be in any case acceptable due to the important advantages of employing a distributed algorithm.

5.4 A different Approach

In section 4, we have seen as both the coverage criteria (O1) and (O2), i.e. the intervention and the visibility problems, can be characterized by two distinct objective functions and so far we have considered such functions only separately. However, in some cases it might be of considerable importance to take into account them at the same time. In general, one cannot simultaneously optimize both functions, unless the functions share common optima. Hence, our idea is to optimize a combined objective function that strikes a compromise between maximizing visible area and minimizing the distance of the robots to points in the environment. By introducing such objective function, we achieve to render the CAO algorithm applicable to the particular problem of 3D multi-robot surveillance coverage treated in this section. Like in the previous cases, this objective function depends on the unknown terrain's characteristics and thus its explicit form is not known. However, for any given team configuration the value of this objective function can always be directly computed from the robots' sensor measurements, and thus the CAO algorithm can be applied to the problem at hand by using such an objective function.

The motivation behind the choice to combine simultaneously both the coverage criteria is that, even if they apparently seem very similar, in many practical situations they may end up to be complementary. Indeed, in several simple conditions they can lead to solution very close one to each other, but in some common cases the final configurations have proprieties

completely different. To better understand this important point, we propose two simple examples. Firstly, let us consider the intervention time as the only criterion for our system and let the region to cover be very complex, e.g. a planar region with many holes (obstacles). In this case an optimal solution for this problem might be a configuration from which the team is able to see only a minor percentage of the environment. On the other hand, let us imagine the visibility problem in a circular region and a robot with a sensor able to monitor at distances greater than the diameter of the circle. In this case every robot position is a solution of the problem. However, we cannot state that all such solutions are equivalent for a surveillance task: attempting to keep the robots as close as possible to the points in the area is necessary for two practical reasons: (a) firstly, the closer is the robot to a point in the terrain the better is, in general, its sensing ability to monitor this point and (b) secondly, in many multi-robot coverage applications there is the necessity of being able to intervene as fast as possible in any of the points of the terrain with at least one robot.

For these reasons, while maximizing the visible area is the primary goal of the mission, the team members should be deployed so that for every point in the terrain, the closest robot is as close as possible to that point. In other words, among all possible configurations that maximize the visible area $\mathcal{V}$, the robot team should converge to the one that keeps as small as possible the average distance between each of the robots and the terrain subarea the particular robot is responsible for, where the subarea of the terrain the i-th robot is responsible for is defined as the part of the terrain that (a) is visible by the i-th robot and (b) each point in this subarea is closer to the i-th robot than any other robot of the team.

Having the above reasoning in mind, we define the following combined objective function the robot team has to minimize:

$$J(\mathcal{P}) = \int_{q\in\mathcal{V}} \min_{i\in\{1,\ldots,M\}} \|x^{(i)} - q\|^2 dq + K \int_{q\in\mathcal{T}-\mathcal{V}} dq \tag{5.1}$$

where K is a user-defined positive constant and $\|\cdot\|$ denotes the Euclidean norm. The first of the terms in the above equation is the usual cost function considered in many coverage problems for known 2D environment related to the second objective (minimize the average distance between the robots and the subarea they are responsible for, see [20]). The second term is related to the invisible area in the terrain ($\int_{q\in\mathcal{T}-\mathcal{V}} dq$ is the total part of the terrain that is not visible by any of the robots).

The positive constant K serves as a weight for giving less or more priority to one of the objectives. For instance, in the case where $K = 0$, we will have that the robots, in their attempt to minimize their average distance to the subarea they are responsible for, may also seek to minimize the total visible area. It has to be emphasized that the positive constant K

should be chosen sufficiently large so that the second term in (5.1) dominates the first term unless no or a negligible part of the terrain remains invisible. In this way, minimization of (5.1) is equivalent to firstly making sure that all, or almost all, of the terrain is visible and then to locate the robots so that their average distance to the subarea they are responsible for is minimized. However, choosing a value for K so that the second term in (5.1) dominates the first term is not straightforward unless the terrain is known. Later in this section we will comment further on how to choose the parameter K for the particular setup considered in this thesis. Also note that whether the CAO-based algorithm employing a large K converges to negligible or non-negligible invisible areas depends on the number, the sensing capabilities and maximum height constraints of the robots as well as the terrain's complexity.

The second term $\int_{q\in\mathcal{T}-\mathcal{V}} dq$ in (5.1) cannot be, in general, computed in practice; as this term involves the part of the terrain that is not currently visible, its computation requires that the geometry of this part is known or equivalently – as the invisible part changes with the evolution of the team's configuration – that the whole terrain is known. To overcome this problem, instead of minimizing (5.1) the following performance index is actually minimized by the CAO approach:

$$\bar{J}(\mathcal{P}) \;=\; \int_{q\in\mathcal{V}} \min_{i\in\{1,\ldots,M\}} \|x^{(i)}-q\|^2 dq - K\int_{q\in\mathcal{V}} dq \tag{5.2}$$

To see that minimization of (5.2) and (5.1) is equivalent, note that

$$\int_{q\in\mathcal{T}-\mathcal{V}} dq = \int_{q\in\mathcal{T}} dq - \int_{q\in\mathcal{V}} dq$$

and the integral over the entire region is constant.

Throughout the above analysis, the assumption that K is "sufficiently large" was made. If such an assumption does not hold, the arguments presented above do not hold either. Therefore, the question that naturally arises is what is a value for K that is sufficiently large so that these arguments hold. As a very large choice for K (e.g., $K = 10^{10}$) can lead to numerical instability problems (switching-like performance for the algorithm when the numerically-computed invisible area switches from small values to zero), a criterion on how to choose K so that such instability problems do not occur should be provided. Extensive simulations with all set-ups considered in the next section (sec. 5.5) and with different values for K indicate that it suffices to choose K to be $3-50$ times the parameter $\hbar_{max}$ in order to get an efficient

performance, where the parameter $\hbar_{max}$ can be calculated as follows: let

$$\begin{aligned} f(\mathcal{P}) &= \int_{q\in\mathcal{V}} \min_{i\in\{1,\ldots,M\}} \|x^{(i)} - q\|^2 dq, \\ \bar{g}(\mathcal{P}) &= \int_{q\in\mathcal{T}-\mathcal{V}} \mathcal{I}(x,y)dxdy \end{aligned}$$

where $\mathcal{I}(x,y)$ denotes the indicator function that is equal to 1 if the point q belongs to the invisible area of the terrain and is zero, otherwise. In other words, the term $\bar{g}(\mathcal{P})$ would correspond to the total invisible area, if the unknown terrain points (x,y,z) were replaced by $(x,y,1)$, i.e., if the whole invisible area were flat. Then the parameter $\hbar_{max}$ is calculated according to

$$\hbar_{max} \approx \frac{\sup f(\mathcal{P})}{\sup \bar{g}(\mathcal{P})}$$

i.e., $\hbar_{max}$ corresponds to the maximum possible value for $f(\mathcal{P})$ (over all possible feasible team configurations) divided by the maximum value the invisible-area-term $\bar{g}(\mathcal{P})$ can take. The $\sup \bar{g}(\mathcal{P})$ is equal to the terrain's area along the (x,y)-axes, i.e. $\sup \bar{g}(\mathcal{P}) = (x_{max} - x_{min})(y_{max} - y_{min})$. On the other hand an estimate of the term $\sup f(\mathcal{P})$ can be produced by running extensive simulations with randomly generated terrains and randomly generated team's configurations. Figure 5.3 shows the time-histories of the terms $f(\mathcal{P})$ and $\bar{g}(\mathcal{P})$ for different choices for K and for one of the scenarios described in the simulations section (more precisely, for the scenario described in section 5.5.2). For this particular scenario, we have that $\sup f(\mathcal{P}) \approx 1000$ and $\sup \bar{g}(\mathcal{P}) = 100$ and thus the parameter $\hbar_{max}$ can be estimated to be around 10. As exhibited in Figure 5.3, the CAO-based algorithm converges to negligible invisible areas for all values of K satisfying $K \in [3\hbar_{max}, 50\hbar_{max}] \equiv [30, 500]$.

5.5 Simulation Results

To evaluate the efficiency of the proposed approach, several scenarios were considered with the use of a simulated robot team which was able to move freely at the 3D plane. In all cases studied, the team was homogeneous with exactly the same monitoring capabilities. This assumption has been made for simplification purposes and easier comprehension of the results. 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 they have to satisfy a maximum height requirement while they do not hit the terrain, i.e., they remain within $[z + d, z_{max}]$ along the z-axis. The scenarios considered are terrains with obstacles with same or uneven heights, while for each scenario different values of the parameter

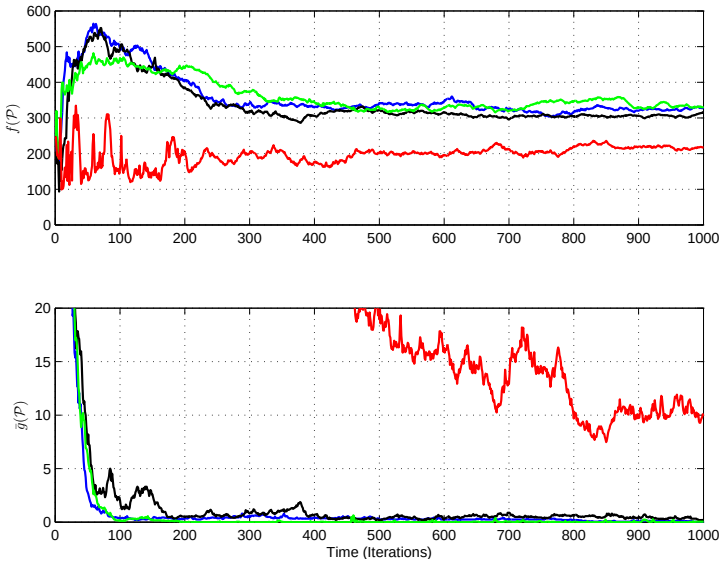


Figure 5.3: Time-histories for the terms $f(\mathcal{P}) = \int_{q\in\mathcal{V}} \min_{i\in\{1,\ldots,M\}} \|x^{(i)} - q\|^2 dq$ (upper plot) and $\bar{g}(\mathcal{P}) = \int_{q\in\mathcal{T}-\mathcal{V}} \mathcal{I}(x,y)dxdy$ (lower plot) for different values of the parameter K: $K = 5$ (red), $K = 30$ (black), $K = 100$ (blue) and $K = 500$ (green).

α which is responsible for the convergence of the algorithm were tested.

In all the experiments here reported, the following choices were made for the algorithm's implementation:

- The CAO parameters N (number of next candidate robots' positions) and L (size of approximator ϕ) were set equal to $6M$ and $6M + 1$, respectively, where M denotes the robot team's size, while the approximator ϕ was calculated as described in the previous section. Note that the above choices for N, L and ϕ are in accordance to Theorem 1 and Remark 5; moreover, L was set equal to $6M + 1$ as it has to be an odd number.

- The parameter K in the cost criterion (5.2) was set equal to 30 which satisfies $K \in [10\hbar_{max}, 50\hbar_{max}]$ for all terrains and team sizes considered in the simulations (see section 5.4 for more details on the parameter $\hbar_{max}$).

- The parameter d (minimum allowable distance from the terrain) was set equal to 0.1, while the robot's were assumed to have unlimited visibility.

- Different choices for the parameters z_{max} (maximum allowable height) and α (magnitude of next candidate robots' positions) were made as these parameters are the most crucial for the algorithm's performance.

Table 5.1: Coverage percentage in the case described in Section 5.5.1.

	(% of Coverage)		
α	0.3	0.5	1
Initial Configuration	34.58		
Final Configuration	97.06	97.49	98.59

- Finally, with the exception of the experiments reported in section 5.5.4 that involve teams of 10 and 20 robots, in all other cases the team comprised 4 robots.

5.5.1 Areas with equal-height obstacles

The first case considered, studies an area sizes 10 by 10 meters, which includes a surface with seven equal-height randomly placed obstacles. For this area, several scenarios were tested with the robot team starting from different initial positions and heights. In all the considered cases the robots had to satisfy a maximum flight height requirement while they did not hit the terrain.

Scenario 1

In the first studied scenario, all the team members were placed at starting points adjunct to each other, with initial height 0.6 meters. The maximum allowed flight height was 1 meter for all robots. Different values of the expression α were tested for the case of $\alpha = 0.3, 0.5, 1$ and the respective cost functions are presented in Fig. 5.4. The initial position for *Robot 1* was $(0.18, 0.2, 0.6)$, of *Robot 2* was $(0.19, 0.2, 0.6)$, of *Robot 3* was $(0.2, 0.2, 0.6)$ and of *Robot 4* was $(0.21, 0.2, 0.6)$. In Fig. 5.5 successive snapshots of different positions of the robot team for the case of $\alpha = 0.3$ are presented (different color corresponds to different team member). The final configuration in all three test cases is presented in Fig. 5.6. In Table 1 the percentage of the initial and final coverage of the area monitored in all three cases, is presented. It's worth mentioning that the coverage percentage is depended on several factors apart the optimization algorithm i.e. the sensors that might be used in a real implementation. It should be noted that CAO does not converge always to the same swarm configuration, but it converges always to a swarm configuration with similar coverage characteristics which corresponds to similar final J value.

Scenario 2

In the second scenario, the initial positions of the robots were for *Robot 1* $(9.18, 0.2, 0.4)$, for *Robot 2* $(9.19, 0.2, 0.4)$, for *Robot 3* was $(0.2, 0.2, 0.4)$ and for *Robot 4* was $(0.21, 0.2, 0.4)$,

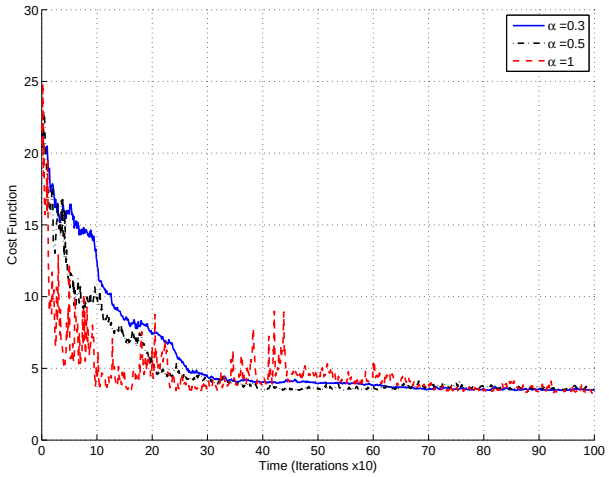


Figure 5.4: Cost Functions for $\alpha = 0.3, 0.5, 1$, in the case described in Section 5.5.1.

while the maximum allowed flight height remained the same (1 meter). Different values of the expression α were tested for the case of $\alpha = 0.3, 0.5$ and the respective cost functions are presented in Fig. 5.8. In table 2 the percentage of the initial and final coverage of the area monitored in both cases is presented.

Table 5.2: Coverage percentage in the case described in Section 5.5.1.

	(% of Coverage)	
α	0.3	0.5
Initial Configuration	48.71	
Final Configuration	98.56	97.04

5.5.2 Areas with uneven obstacle height

The second considered case, studies an area sizes 10 by 10 meters, which includes a surface with seven randomly placed obstacles with uneven height, with maximum value 2 meters. In this test case we have tested several scenarios with the robot team starting from different initial positions and heights. In all the considered cases the robots had to satisfy a maximum flight height requirement while they did not hit the terrain.

Scenario 1

In the first studied scenario for the case of areas with uneven obstacle heights, all the team members were placed at starting points adjunct to each other, with initial height 0.2 meters.

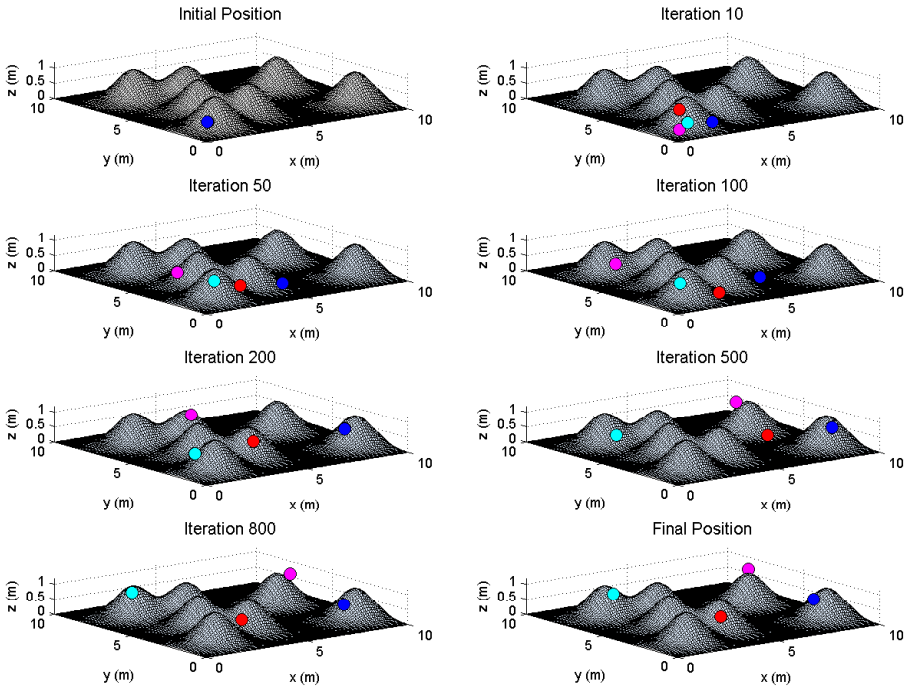


Figure 5.5: Successive snapshots of different positions of the robot team for $\alpha = 0.3$, in the case described in Section 5.5.1.

The maximum allowed flight height was 1 meter for all robots. The initial positions of *Robot 1* was $(0.18, 0.2, 0.2)$, of *Robot 2* was $(0.19, 0.2, 0.2)$, of *Robot 3* was $(0.2, 0.2, 0.2)$ and of *Robot 4* was $(0.21, 0.2, 0.2)$. Different values of the expression α were tested for the case of $\alpha = 0.3, 0.5, 1$ and the respective cost functions are presented in Fig. 5.9. The final configuration in all three test cases is presented in Fig. 5.10, while in Table 3 the percentage of the initial and final coverage of the area monitored in all cases, is presented.

Table 5.3: Coverage percentage in the case described in Section 5.5.2.

	(% of Coverage)		
α	0.3	0.5	1
Initial Configuration	29.78		
Final Configuration	98.29	97.76	96.35

Scenario 2

In the second studied scenario for uneven surfaces, the maximum allowed flight height was 5 meters for all robots. Different values of the expression α were tested for the case of

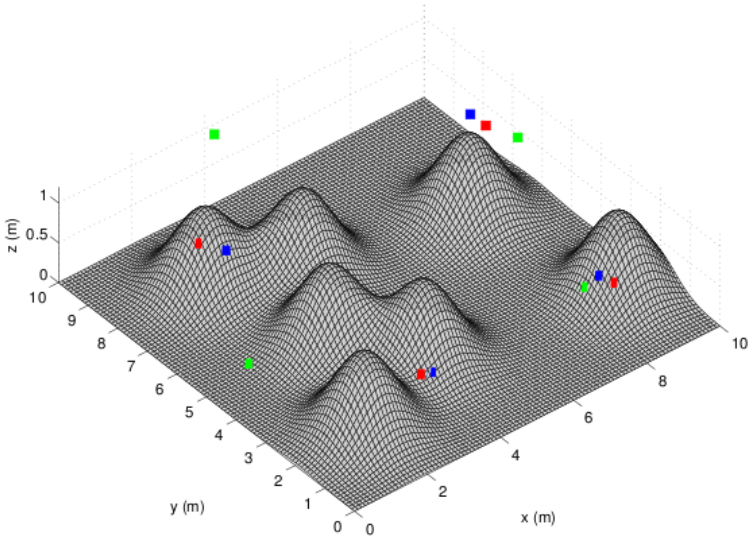


Figure 5.6: Final positions of the robotic teams for $\alpha = 0.3$ (blue markers), $\alpha = 0.5$ (red markers), $\alpha = 1$ (green markers), in the case described in Section 5.5.1.

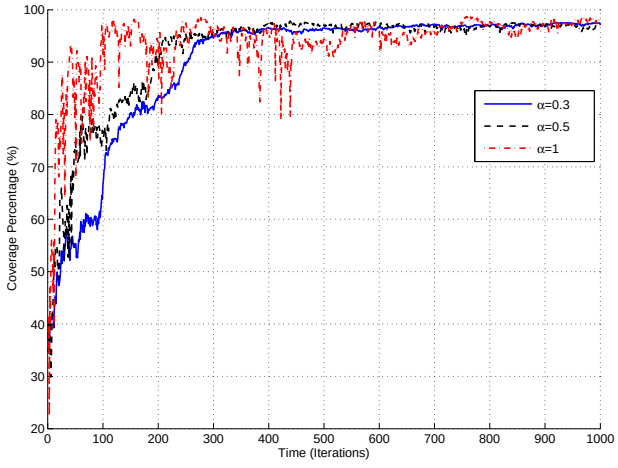


Figure 5.7: Coverage percentage for $\alpha = 0.3, 0.5, 1$, in the case described in Section 5.5.1.

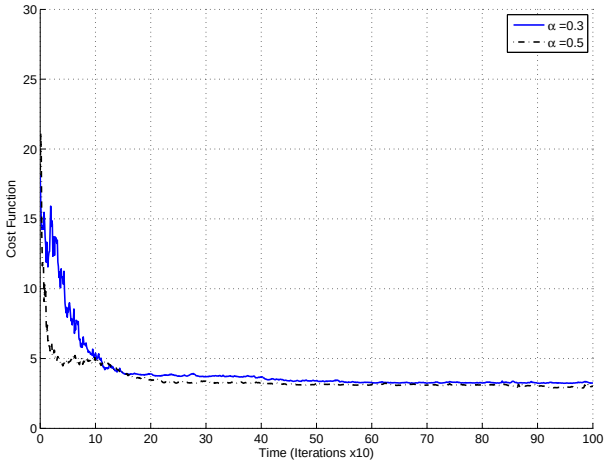


Figure 5.8: Cost Functions for $\alpha = 0.3, 0.5$, in the case described in Section 5.5.1.

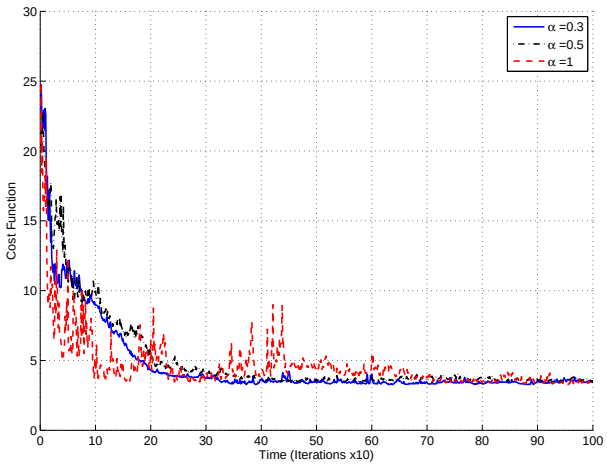


Figure 5.9: Cost Functions for $\alpha = 0.3, 0.5, 1$, in the case described in Section 5.5.2.

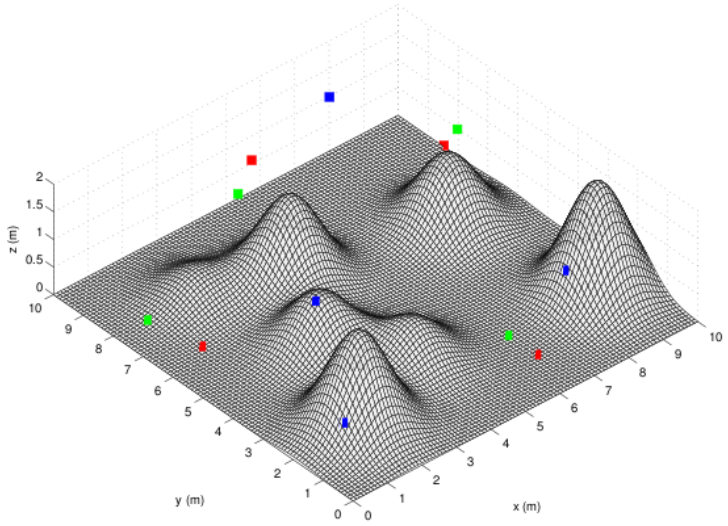


Figure 5.10: Final positions of the robotic teams for $\alpha = 0.3$ (blue markers), $\alpha = 0.5$ (red markers), $\alpha = 1$ (green markers), in the case described in Section 5.5.2.

$\alpha = 0.3, 0.5, 1$ and the respective cost functions are presented in Fig. 5.11. In Fig. 5.12 successive snapshots of different positions of the robot team for the case of $\alpha = 0.3$ are presented (different color corresponds to different team member). The initial position of *Robot 1* was $(0.18, 0.2, 0.4)$, of *Robot 2* was $(0.19, 0.2, 0.4)$, of *Robot 3* was $(0.2, 0.2, 0.4)$ and of *Robot 4* was $(0.21, 0.2, 0.4)$. In table 4 the percentage of the initial and final coverage of the area monitored in all cases, is presented.

Table 5.4: Coverage percentage in the case described in Section 5.5.2.

	(% of Coverage)		
α	0.3	0.5	1
Initial Configuration	29.78		
Final Configuration	99.14	98.69	98.36

5.5.3 Cave-like Surface

In the above described set-up and proposed simulations, for simplicity's sake we assumed that the unknown terrain is defined as a set of unique triplets (x, y, z), that is, for each (x, y) the terrain is defined by a unique z-point, i.e. $z = f(x, y)$. As already sentenced, the proposed method is able to deal with any kind of terrain morphology and it is a crucial propriety since in realistic applications there exist several cases where there may be more than one z-points (e.g., cases of terrains with buildings, overhangs, ledges, caves, etc). Here we present a similar

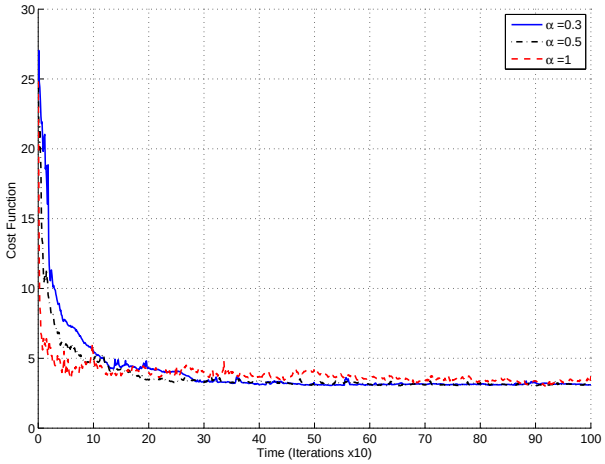


Figure 5.11: Cost Functions for $\alpha = 0.3, 0.5, 1$, in the case described in Section 5.5.2.

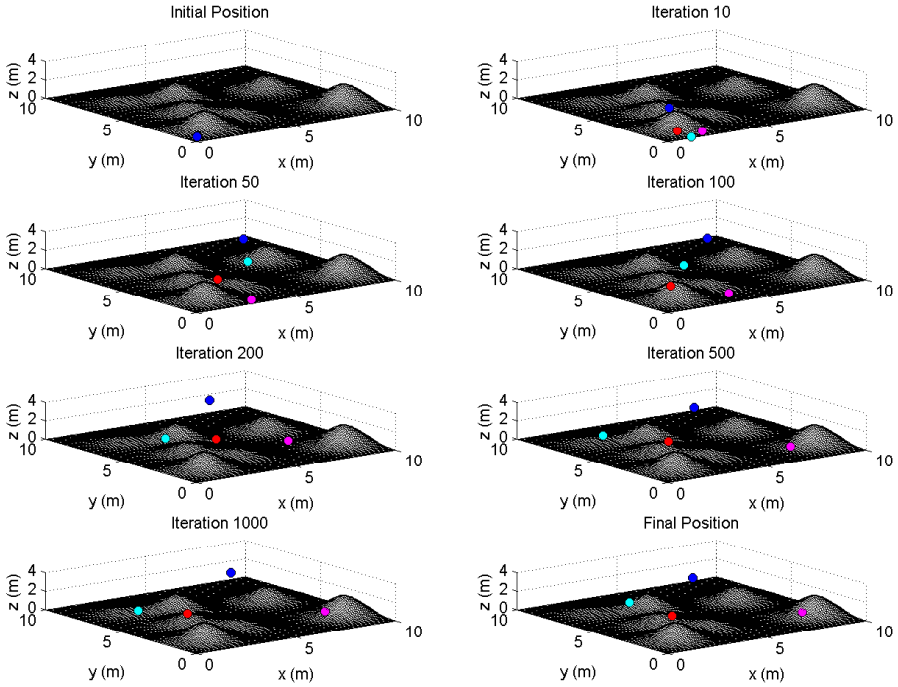


Figure 5.12: Successive snapshots of different positions of the robot team for $\alpha = 0.5$, in the case described in Section 5.5.2.

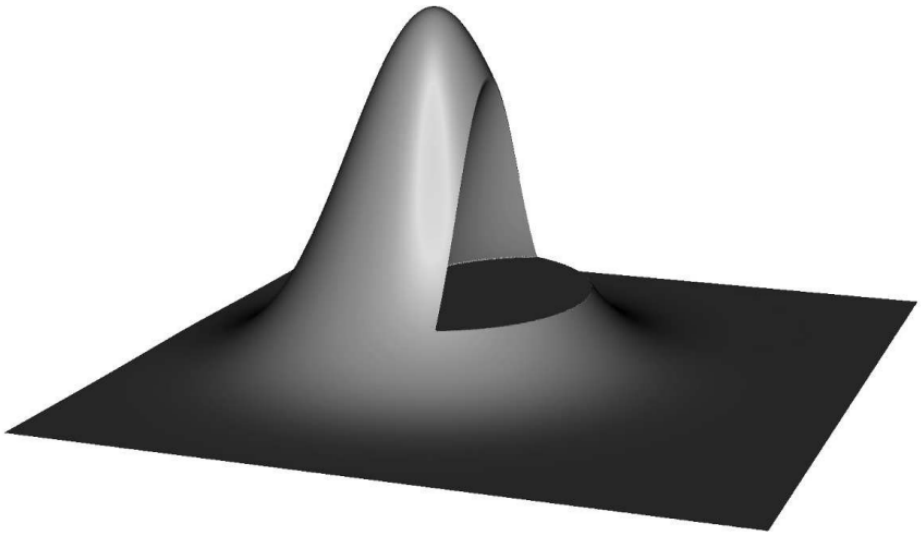
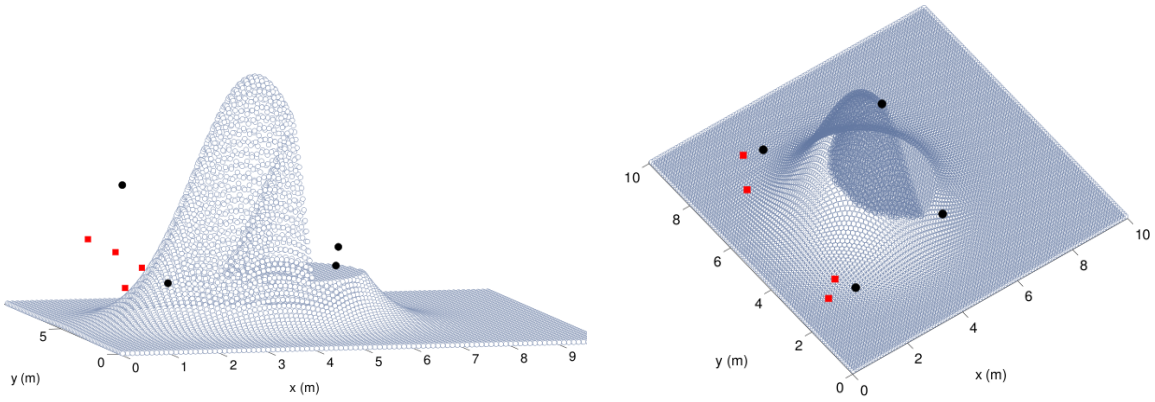


Figure 5.13: A different scenario: the environment is a gaussian with a cave.

scenario to show how our method can be applied also for these cases. The simulated environment is a gaussian with a cave (Fig. 5.13). The robots start their mission on the other side with respect to the cave, so at the beginning it is not visible. Start and final robots' positions are shown in Fig. 5.14. In Fig. 5.15(a) the behavior of the cost function is presented and in Fig. 5.15(b) it is shown also the percentage of the invisible surface during the task. It is possible to see that it is minimized until everything is visible.

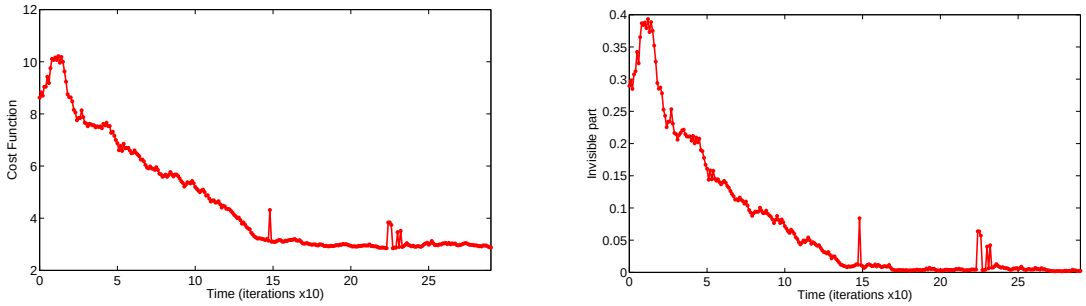
5.5.4 Scalability Issues

To validate the efficiency in the case of bigger robot teams, we have performed experiments with teams consisting of 10 and 20 members. Our experiments were performed in an area sizes 20 by 20 meters, which includes a surface with fifteen uneven height randomly placed obstacles. The maximum flight height was set to be 2. The basic difference as far as it concerns the computational requirements in the experiments conducted with the teams of 10 and 20 robots, was that the parameters L and N increase linearly according to Theorem 1 and Remark 5; therefore $L = 61$ and $N = 60$ in the case of the team with 10 members and $L = 121$ and $N = 120$ in the case of the team with 20 members. In the case of 4 robots the best values of $\mathcal{J}$ are around 15 which is significantly larger than the values obtained with the bigger teams. In Fig. 5.16 the cost functions for the case of 10 and 20 robots are presented, while in Fig. 5.17 we present their final configuration. In table 5 the percentage of the initial and final coverage of the area monitored for a team with 10 and 20 members is presented.



(a) (b)

Figure 5.14: Coverage mission in an environment with a cave. The team is composed by four robots. Red squares and black circles represent initial and final positions respectively. At the beginning the robots cannot see the cave.



(a) (b)

Figure 5.15: Fig. (a) shows the behavior of the cost function during the task, Fig. (b) the percentage of invisible surface. At the end everything is visible.

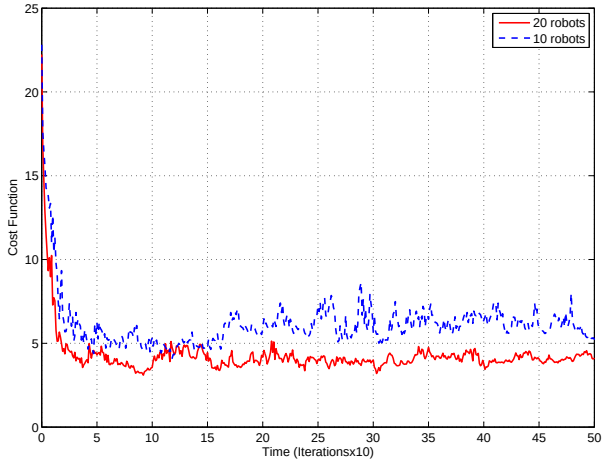


Figure 5.16: Comparative cost functions for the case of a robot team with 10 and 20 members.

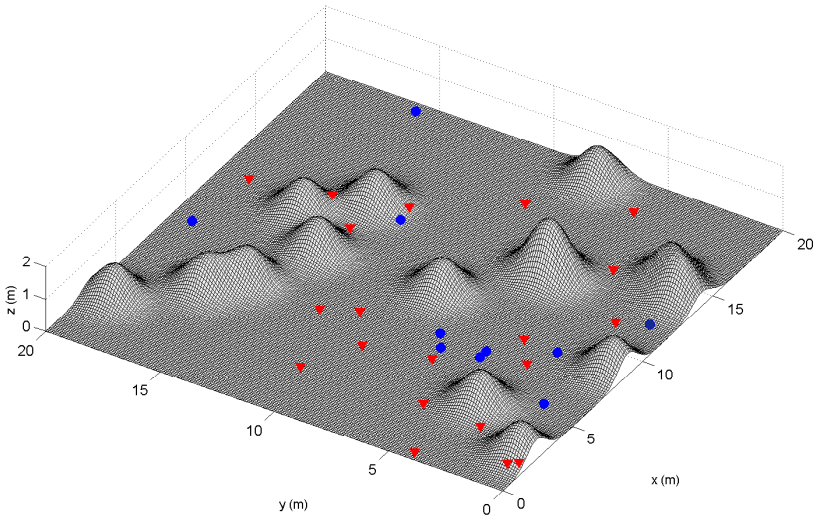


Figure 5.17: Final configuration of the team with 10 robots (blue circle markers) and the team with 20 robots (red triangle markers).

Table 5.5: Coverage percentage with teams of 10 and 20 members.

	(% of Coverage)	
Team size	10 Members	20 Members
Initial Configuration	39.62	42.93
Final Configuration	86.33	90.78

5.6 Conclusions

In this chapter we extended the results presented in Chapter 4 for the coverage of a surface in a 3D environment. Firstly, we straightly extended the distributed solution provided in the previous chapter, showing that it can be applied also in this case. We have also compared these results with those obtained by using the centralized approach. The comparison was completely satisfying since the two methods' performances are very similar. Then, we introduced a new possible objective function which tries to take into account both the coverage criteria previously introduced: the intervention problem and the visibility problem. In general, one cannot simultaneously optimize both functions, unless the functions share common optima. Hence, the idea is to optimize a combined objective function that strikes a compromise between maximizing visible area and minimizing the distance of the robots to points in the environment. The application of our algorithm to this problem was presented and a performance evaluation by means of several simulations, performed by using complex simulated environments, was provided. Additionally, some results to prove the scalability of the algorithm with respect to the number of robots are presented. The obtained results show that the CAO algorithm is able to provide a solution for the cooperative surveillance coverage for a completely arbitrary 3D environment. After the validation by using simulated environment, in the next chapter we present the experimental results where the algorithm is tested using real data provided by a swarm of MAVs for a real coverage mission.

Chapter 6

Experimental Results

In this chapter we present an important part of our work: the integration of our work with the experimental results obtained by using a real swarm of MAVs. This part of work is in collaboration also with the other partners of the sFly project and in particular with the ETH of Zurich (see [23], [24]).

Firstly, we focus our study on the implementation of a two-step procedure which allows us to align optimally a team of flying vehicles for a surveillance task. Initially, a single robot constructs a map of the area of interest using a novel monocular-vision-based approach. A state-of-the-art visual-SLAM algorithm tracks the pose of the camera while, simultaneously and autonomously, building an incremental map of the surrounding environment. The generated map is processed and serves as an input for the CAO algorithm. The output of this procedure is the optimal arrangement of the robot team, which maximizes the monitored area. The efficiency of our approach is demonstrated using real data collected from aerial robots in different outdoor areas.

A key issue for the successful implementation of the CAO proposed methodology in the case of a team of MAVs, is the accuracy of the input, which in this case is an elevation map of the environment. We consider an elevation map as a trade-off between complex environmental mapping versus online availability of the environment shape for real-time coverage. A more sophisticated, yet much more costly approach in terms of computational complexity, is presented in [116]. There, the authors reconstruct the 3D environment with the aid of Multi Level Surface maps on a ground robot. Since MAVs generally fly at a reasonable altitude, the area is well approximated by a computationally much less expensive elevation map not considering tunnels or cave-like structures.

Since we deal with MAVs, the choice of sensors to perceive the environment to be monitored and therefore to construct the elevation maps is limited. For GPS-denied navigation

and mapping, vision sensors and laser range finders might be the only options. In [67] the authors combine range images with a digital elevation model for accurate environment modeling. A computationally less complex approach was chosen by [115] using a multi-resolution approach adopted from the computer graphics literature. This approach shows real-time capabilities on a ground robot. However, in aerial navigation, we have even more constraints in the computation power budget. Furthermore, laser scanners are too heavy for MAVs and have a limited field of view. Therefore, cameras and inertial sensors might be the only viable solutions for such limited weight and calculation power budgets. For ground vehicles (cars), 3D occupancy grids built from stereo vision and GPS data have been shown to be a valid solution [18]. However, occupancy grids are not a good option for MAVs because of their limited calculation power. Lacroix [101] presented an off-line method to map a large outdoor scenario in fine resolution using low-altitude aerial stereo-vision images. However, stereo vision loses its advantage when the baseline is too small compared to the scene depth. Considering the limited weight, power and computation budget on MAVs, we rely on a monocular solution in which the appropriate baseline is provided by a keyframe-based visual SLAM framework [40].

6.1 Platform

For completeness, we provide here a description of the experimental platform used for the final implementation considering both hardware and software aspects [24].

6.1.1 Hardware

The MAV we use is a so-called quadrocopter, a helicopter driven by for rotors, symmetric to the center of mass. The control of the quadrocopter is performed solely by changing the rotation speed of the propellers and is described in more details in [38]. For our experiments, we use the "AscTec Pelican" quadrocopter [36], which is a further development of the one described in [38]. The quadrocopter is equipped with rotors with 10" diameter which allow to carry a payload of about $500g$. Depending on battery size and payload, flight times between 10 and 20 minutes can be achieved. Further key features are the Flight Control Unit (FCU) "AscTec Autopilot" as well as the flexible design enabling one to easily mount different payloads like computer boards or cameras. The FCU features a complete Inertial Measurement Unit (IMU) as well as two $32Bit$, $60MHz$ ARM-7 microcontrollers used for data fusion and flight control. One of these microcontrollers, the Low Level Processor (LLP) is responsible for the hardware management and IMU sensor data fusion. An attitude and GPS-based po-

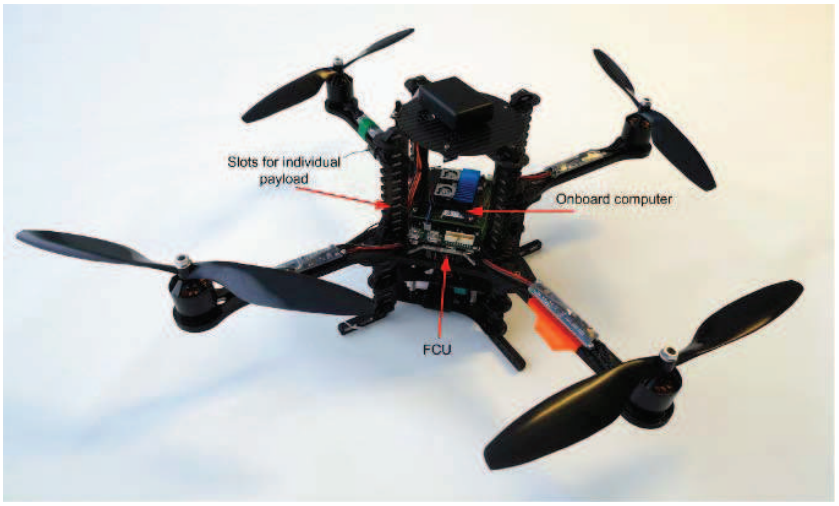


Figure 6.1: Overview of the Pelican quadrocopter.

sition controller is implemented as well on this processor. The LLP is delivered as a black box with defined interfaces to additional components and to the High Level Processor (HLP). To operate the quadrocopter, only the LLP is necessary. Therefore, the HLP is dedicated for custom code. All relevant and fused IMU data are provided at an update rate of $1kHz$ via a highspeed serial interface. In particular, this comprises body accelerations, body angular velocities, magnetic compass, height measured by an air pressure sensor and the estimated attitude of the vehicle.

For the computationally more expensive onboard processing tasks, we outfitted the helicopter with a 1.6 GHz Intel Atom Based embedded computer, available from [36]. This computer is equipped with 1 GB RAM, a MicroSD card slot for the operating system, a 802.11n based miniPCI Express WiFi card and a Compact Flash slot. The miniPCIE WiFi card is preferred over USB to keep the USB bus free for devices like the cameras we use. We furthermore use a high speed CF-card that allows us data logging with up to 40 MByte/s.

As camera, we use a Point-Grey USB Firefly camera with a resolution of $752 \times 480px$ and a global shutter. The camera faces the ground with a $150°$ field-of-view lens since we are expecting the most stable features trackable over longer time in this configuration.

The configuration of our system is schematically depicted in Figure 6.2.

6.1.2 Software

To provide a maximum portability of our code and to avoid potential (binary) driver issues, we installed Ubuntu Linux 10.04 on our onboard computer which makes tedious crosscompiling

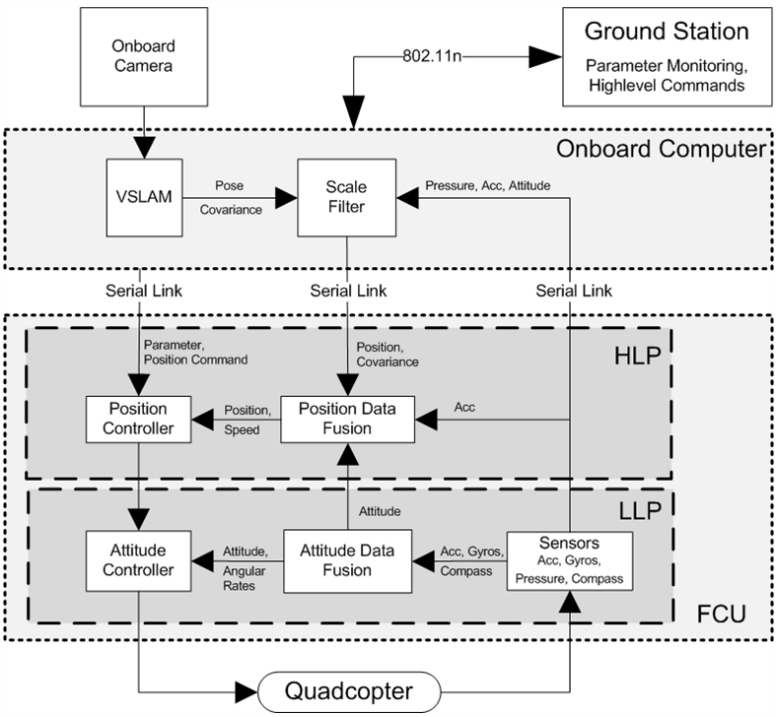


Figure 6.2: Overview of the onboard schematics and interfaces.

unnecessary. Since we are running a couple of different subsystems that need to communicate one each other, we use the ROS (Robot Operating System from Willow Garage) [88] framework as a middleware. This is also used to communicate to the ground station over the WiFi data-link for monitoring and control purposes. The FCU is interfaced via a ROS node communicating over a serial link to the FCU's Higlevel Controller with firmware we developed for our purposes. Software development on the HLP is done based on a SDK available for the AutoPilot FCU providing all communication routines to the LLP and a basic framework. The HLP communicates with the ROS framework on the onboard computer over a serial datalink and a ROS FCU-node handling the serial communication. This node subscribes to generic ROS pose messages with covariance, in our case from the vision framework, and forwards it to the HLP. Moreover, it allows to monitor the state of the fusion filter and the position controller, and to adjust their parameters online via the "dynamic reconfigure" functionality of ROS.

For the implementation of the position control loop and data fusion onboard the HLP, a Matlab/Simulink framework is used in combination with the Mathworks Real-Time Workshop Embedded Coder. The framework provides all necessary tools to design the control structure in Simulink, optimize it for fixed point computing, as well as compiling and flashing the HLP.

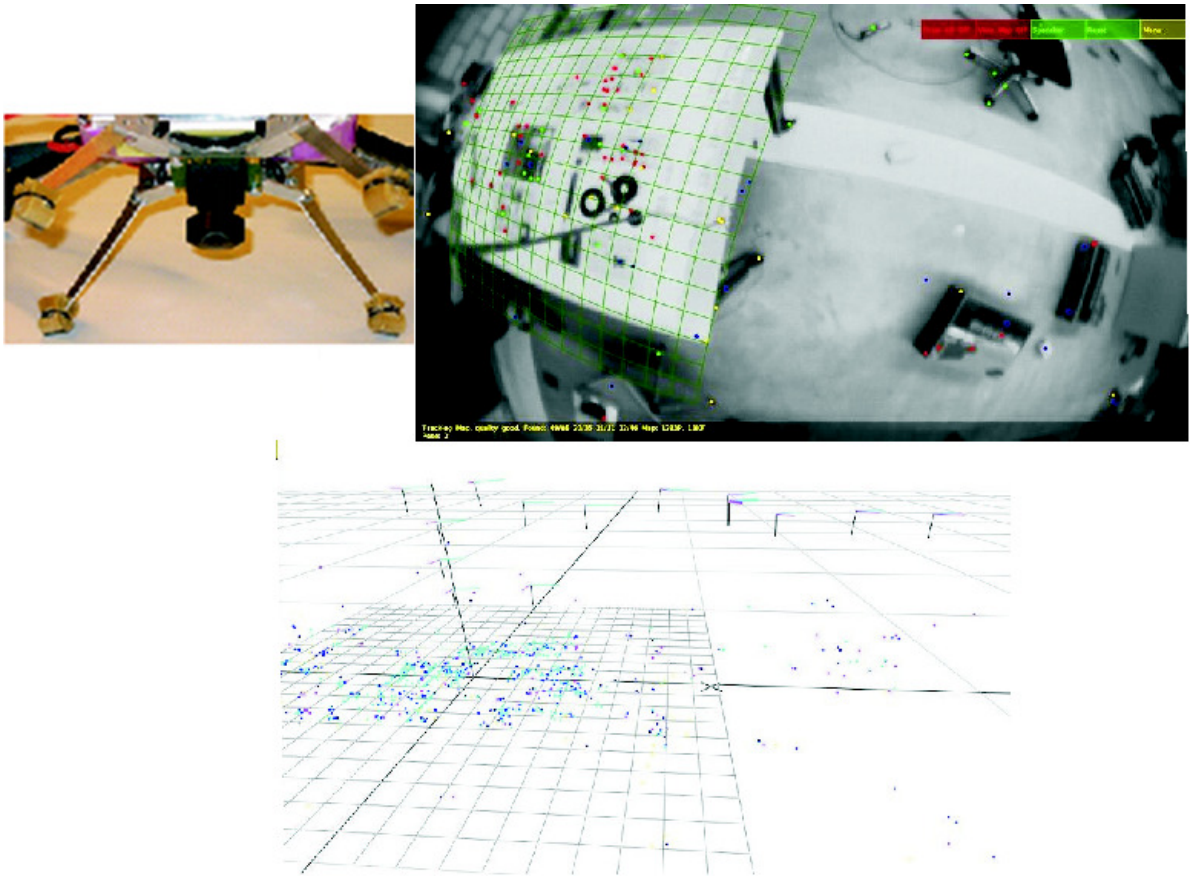


Figure 6.3: The top-left picture depicts the onboard-mounted camera on our vehicle (the Pelican) from Ascending Technologies. The top-right picture is a screenshot of our visual SLAM algorithm. The tracking of features can be observed. This is used for the localization of the camera. In the bottom picture, the 3D point cloud map built by the mapping thread is shown. The 3-axis coordinate frames represent the location where new keyframes were added.

6.1.3 Mono-Vision Framework

The approach here presented uses the keyframe based visual SLAM algorithm of Klein and Murray [59] in order to localize the MAV and build a dense elevation map with a single camera.

In summary, Klein and Murray split the simultaneous localization and mapping task into two separately scheduled threads: the tracking thread and the mapping thread. The tracking thread is first of all responsible for the tracking of salient features in the camera image, i.e., it compares the extracted point features with the stored map and thereby attempts to determine the position of the camera. This is done with the following steps: first, a simple motion model is applied to predict the new pose of the camera. Then the stored map points are projected into the camera frame and corresponding features (FAST corners in this case) are searched.

This is also referred to as the data association procedure. When this is done, the algorithm refines the orientation and position of the camera such that the total error between the observed point features and the projection of the map points into the actual frame is minimized. The Mapping thread uses a subset of all camera images - also called keyframes - to build a 3D point map of the surroundings. The keyframes are selected using some heuristic criteria. After adding a new keyframe, a batch optimization is applied to the joint state of map points and keyframe poses. This attempts to minimize the total error between projected map points and the corresponding observations in the keyframes. In the computer vision community, this procedure is also referred to as bundle adjustment. It is alternately applied to the global or to a local set of map points and keyframes

When moving the helicopter through a region, our camera is facing downwards. This increases the overlapping image portion of neighboring keyframes, so that we can even further loosen the heuristics for adding keyframes to the map. It also ensures, that we can assume an elevation map later on in the meshing procedure. When exploring new areas the global bundle adjustment can be very expensive, limiting the number of keyframes to a few hundred on our platform. An intricate hurdle when using a monocular camera is the lack of any depth information. Closely linked to this problem is the unknown map scale. We tackle this issue with the approach presented in [121] using an inertial sensor. We are thus able to have all distance in metric units.

Adaptations to the SLAM Algorithm

The most evident and crucial change is the importation of the SLAM algorithm to ROS. It facilitates the transport of information to different nodes and computers. From the performance point of view, the most important change is the degeneration of the SLAM framework to a visual odometry framework: We do not keep anymore all keyframes in the bundle adjustment step, but only keep a constant number of them. This makes the algorithm scalable to large environments while keeping the calculation complexity linear with the number of features. If the number of keyframes exceeds a threshold, we only take the closest N keyframes to the current MAV pose. The augmentation in drift is minimal, since keyframes far away from the current MAV pose only contribute minimally in a global bundle adjustment step. Loop closure is handled passively equally to the original version of the algorithm. That is, if the loop did not drift significantly, the keyframe which closes the loop is considered as neighbor of the current MAV pose and is taken into account in the local bundle adjustment step.

Besides the fundamental changes mentioned above, we also adapt some parameters of Klein and Murray's visual SLAM algorithm to increase its performance within our frame-

work for optimal coverage in unknown terrain. First, we use a more conservative keyframe selecting heuristic in order to decrease the number of keyframes added during map expansion. Additionally, we reduce the number of points being tracked by the tracking thread from 1000 to 300. This again increases the maximal map size and the frame rate, while keeping the accurate tracking quality. This leads to a very sparse information for the elevation mesh map, however, our tests show still very satisfying results underlining the strength of our approach for dense elevation mesh maps.

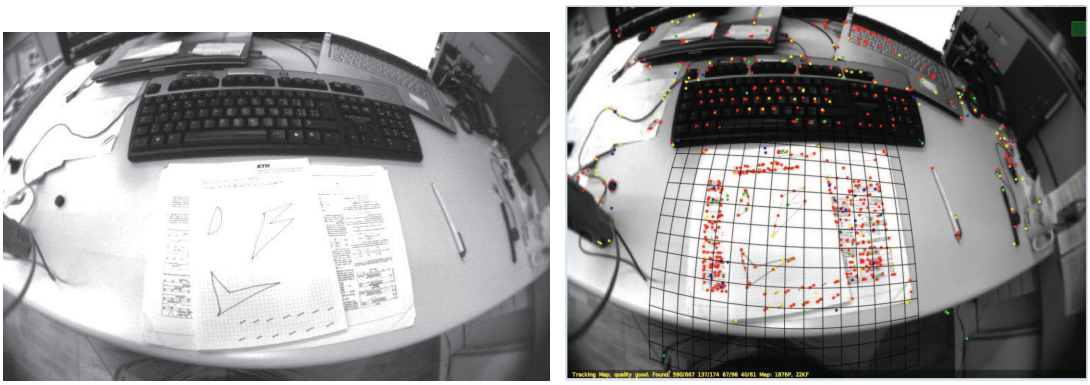
These modifications led to a framerate of max 20Hz on an Intel ATOM 1.6GHz processor. The demonstration of the pure navigation task (i.e. without mesh mapping the environment) is in [120].

6.2 Online Elevation Mesh Map Generation

To perform optimal surveillance coverage over an arbitrary terrain, we need to reconstruct the area in an elevation map. Note that most works on optimal coverage assume an existing map. In this work, we use an approach to build an elevation map online and in real-time. Thus, the MAV has to be able to fly autonomously in the yet-unknown and later-mapped area. For the vision-based autonomous navigation, we use the approach described in section 6.1. We extended the meshing approach of [119] to meet the needs for optimal surveillance coverage in an arbitrary terrain. In particular, we build the map iteratively while the MAVs are exploring the environment. Since we degenerated the visual framework to a visual odometry setup (c.f. section 6.1.3) only the features triangulated with the newest keyframe are added to the meshing process. Notice that, thanks to this modification, the meshing process has constant complexity, since the number of added features per keyframe does not grow with the map size. Furthermore, the required rate of the mesh update is given by the rate of newly added keyframes. That is, it is dependent on the speed/altitude ratio the MAV moves. I.e. the rate of newly added keyframes is the same if the MAV moves fast at high altitude or slower on low altitude. It is the pixel change in the image that triggers a new keyframe. During all our experiments, we use a down-looking camera on the MAV. Thus, we can assume the the point-cloud to be an elevation map.

6.2.1 Elevation Mesh Generation from a Point Cloud

We summarize here the idea presented in [119] for the mesh-map creation. For the sake of simplicity and for better understanding we use a sample scene throughout this section. Figure 6.4 depicts this scene. Note that it is a small scale scene, however, due to our monocular



(a) (b)

Figure 6.4: a) Sample image of the scene mapped for the following illustration of the algorithm in this section. The sheets in front of the keyboard are flat and represent the main plane H whereas the keyboard has a soft inclination in depth towards the upper part of the image. b) Scene with 3D point features. This represents the data available in a keyframe of the visual SLAM algorithm. Back projecting a 3D triangle of the meshed map allows getting the texture for the triangle in question. Note that this is the distorted image while for texturing the mesh we use the undistorted one.

approach, all techniques and algorithms applied to this scene are perfectly scalable. That is, huge terrain captured from far away looks identical to a small terrain captured from very close - i.e. the images and thus the map are scale invariant. At the end of this section we show our algorithm performing in a large scale outdoor environment. Figure 6.4(b) shows the information available in a keyframe of the SLAM algorithm.

Assume the point cloud $\{\vec{p_i}\}$ with M 3D points $\vec{p_i}$ representing the initial map constructed by the visual SLAM algorithm in the start phase. Without any restrictions to the terrain to explore later on we assume the start area to be relatively flat and the aerial vehicle in hover mode. The main map plane H is found using a least square method on $\{\vec{p_i}\}$ or a RANSAC algorithm. In our case the latter one is used to be more robust against outliers. This is done in the given SLAM framework. All current and future map points are projected to this main plane to reduce the dimensionality:

$$\vec{r_i} = P * \vec{p_i} \tag{6.1}$$

where $\vec{p_i}$ is a three dimensional point of the current map and $\vec{r_i}$ is its two dimensional counterpart projected to the main map plane H using the 2x3 projection matrix P. Note that H usually corresponds to a physical plane in the scene (i.e. table or floor). Furthermore, as the camera is down looking on a helicopter this plane usually is only slightly inclined to the xy-

plane in the camera frame. Thus the two dimensional positions of the features $\vec{r_i}$ are accurate while the third (eliminated by the projection) is very noisy due to the depth triangulation of the visual SLAM algorithm. After the projection a Delaunay Triangulation is run in 2D space to generate a 2D mesh. We use a Sweep algorithm for the triangulation to keep calculation power low. For the Sweep triangulation, calculation is in the order of $O(n \log n)$ compared to the standard algorithm with $O(n^2)$. The 3D point cloud of the scene is depicted in Fig. 6.5. One can note the difficulty even a trained eye has to interpret the scene. Standard path planning and obstacle avoidance algorithms cannot be used. In Figure 6.6 the generated mesh is shown. After the Delaunay Triangulation in 2D space we add again the third dimension. As equation (6.1) is not invertible (P is not a square matrix and we therefore have ambiguities in the back projection) we only use the edge information of the Delaunay Triangulation. That is if an edge in the 2D Delaunay Triangulation is defined by

$$d_{2d} = \overline{\vec{r_i}\vec{r_j}} \tag{6.2}$$

we map it to an edge in 3D space according to

$$d_{3d} = \overline{\vec{p_i}\vec{p_j}} \tag{6.3}$$

with $\vec{r_k} = P * \vec{p_k}$ and $k \in$ map. This initial elevation mesh is then median filtered in the third coordinate to remove outliers and noise. The median value is calculated using all adjacent vertices to the center vertex. That is

$$p_{zk} = \text{median}(p_{zi} \forall p_{zi} \in d_{3d} = \overline{\vec{p_k}\vec{p_i}}) \,, \tag{6.4}$$

where p_{zi} denotes the third coordinate of the 3D point $\vec{p_i}$ previously eliminated for the Delaunay Triangulation.

At this point standard path planning and obstacle avoidance algorithms could be applied for enhanced autonomous navigation. The most simple rule for obstacle avoidance is to not traverse the mesh. That is, if the airborne vehicle always stays on the same side of the mesh it will not crash against an obstacle. Note that thanks to the sparseness of the point features this rule is highly robust, however, may be too restrictive in some particular cases. In the task of optimal coverage, we are more interested in the general shape of the landscape, rather than detailed 3D reconstruction. Thus, for the use of optimal coverage, the level of details of these elevation mesh maps is largely sufficient. Note that we can recover the absolute scale factor of the monocular SLAM by using an inertial sensor as we described in [121]. This way, we can reconstruct a metric mesh-map of an arbitrary terrain. Figure 6.7 shows the initialization

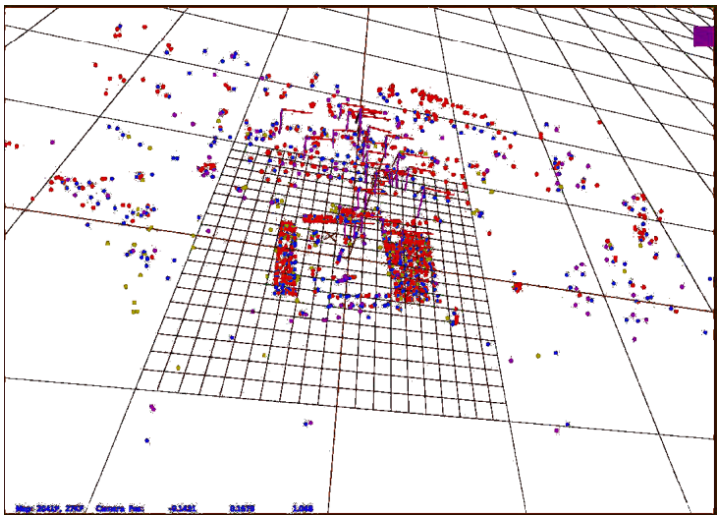


Figure 6.5: The 3D point cloud of the sample scene. A trained eye can spot the papers and the keyboard. However, usually neither human users nor standard path planning and obstacle avoidance algorithms understand the point cloud

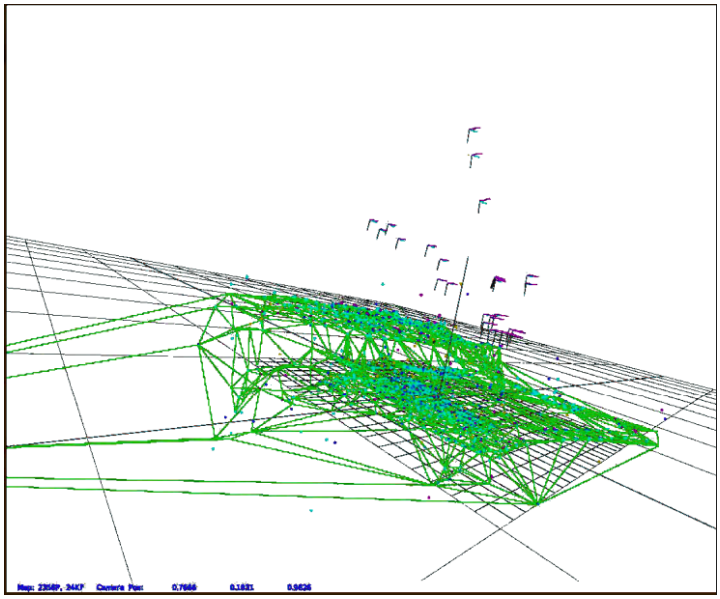


Figure 6.6: Applying Delaunay Triangulation to the point cloud reveals the real topology of the scene. The 'hill' represents the keyboard in the sample scene. Note that we applied a median filter to the mesh vertices in order to eliminate outliers. Thus the 3D points may not always lie on the grid. This grid is already sufficient for path planning and obstacle avoidance.

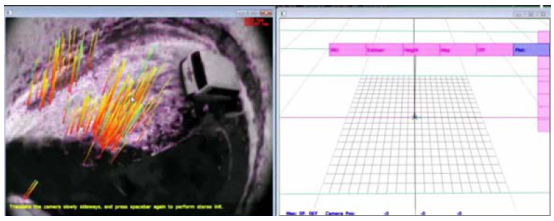
of the visual SLAM algorithm and the reconstruction of our outdoor test terrain. For better visibility we added texture to the mesh map as described in [119]. With the above described procedure, we are able to reconstruct metrically any environment autonomously given that sufficient (arbitrary) visual features are available. In unprepared outdoor environments, this requirement is generally fulfilled. The reconstructed mesh map of the environment can then be used by any coverage algorithm.

6.2.2 Birmensdorf and Hospital area

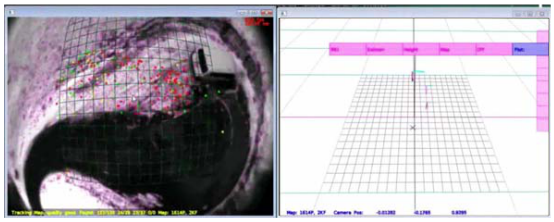
To validate our approach in a realistic environment, we used the data which were collected with the use of the quadrotor previously described. The tested scenarios consider a team of four MAVs and correspond to two different areas. The first area is Birmensdorf in Switzerland and it's presented in Fig. 6.8, while the second area corresponds to the ETHZ's hospital area and it's presented in Fig. 6.9. More details about the data and the methodology used to extract them, are presented in [12] and [119].

In the simulations, 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 they have to satisfy a maximum height requirement while they do not "hit" the terrain, i.e. they remain within $[\Phi(x, y) + d, z_{max}]$ along the z-axis. Several initial configurations for each scenario were tested. The values of the cost function for three different configurations, in the case of the Birmensdorf area are presented in Fig. 6.10. Sample trajectories for a robot team with initial coordinates for *Robot 1* $(1.34, 121.29, 22.91)$, for *Robot 2* $(2.69, 121.29, 22.91)$, for *Robot 3* $(4.04, 121.39, 22.91)$ and for *Robot 4* $(5.39, 121.29, 22.91)$ *(all units are in meters)* are presented in Fig. 6.11, while in Fig. 6.12 the final positions of 3 robot teams starting from different initial positions are presented in a 3D view. Different marker type corresponds to different robots, while different color corresponds to a different team. In table 6.2.2 the final coverage percentage for different initial configurations in the Birmensdorf area, is presented. The values of the cost function for three initial configurations in the case ETHZ's hospital area are presented in Fig. 6.13. Sample trajectories for a robot team with initial coordinates for *Robot 1* $(2.33, 95.57, 41.95)$, for *Robot 2* $(25.64, 97.90, 41.95)$, for *Robot 3* $(48.95, 100.23, 41.95)$ and for *Robot 4* $(72.26, 102.56, 41.95)$ *(all units are in meters)* are presented in Fig. 6.14. In Fig. 6.15 the final positions of 3 robot teams starting from different initial positions are presented in a 3D view.

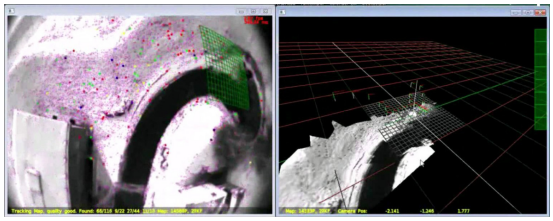
To further validate the efficiency of the proposed methodology, an incremental scenario is also presented. A single aerial robot is flying over an unknown area and incrementally is producing maps which are used as an input to the proposed CAO algorithm. Each increment



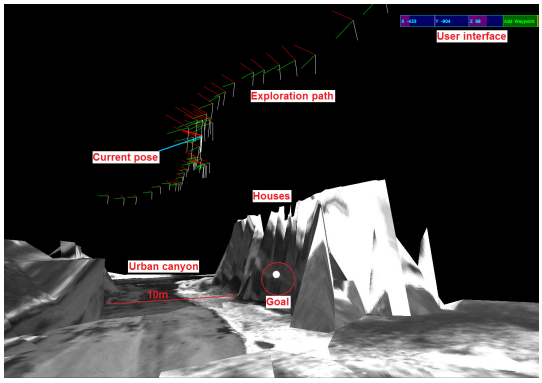
(a)



(b)



(c)



(d)

Figure 6.7: Initialization of the visual SLAM algorithm and reconstruction of our outdoor test terrain. (a) Initialization of the visual SLAM algorithm (on the left the tracked features used to initialize the map, on the right the reference frame). (b) The reference frame is displayed as a grid on the image (left). On the right, a few reconstructed camera poses are displayed as faint tripods. The bold tripod is the actual camera pose. This pose is used for the MAV position controller and yields the metric map scale by fusing it with the IMU measurements. (c) Generation of the textured map. (d) Sample of a meshed and also textured (snowy) outdoor environment. For the CAO approach the generated mesh is sufficient, however, the texture gives the user intuitive information of where the MAV is positioned at the given time instance. Even with the texturing, this approach runs in real-time. Note that the reconstruction precision is not very high. It is, however, largely sufficient for our optimal coverage tasks. With the aid of the IMU we have a metric map and estimate here the urban canyon width to be about 10m (error is $<10\%$). The map reconstruction runs online while flying.

Figure 6.8: Outdoor flight path through the Birmensdorf area. The boundary of the region of interest is in black.

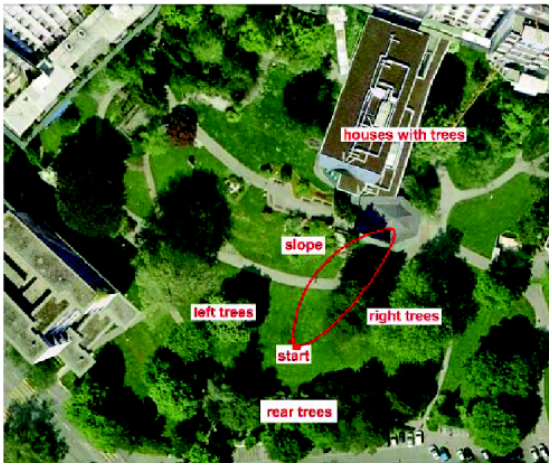


Figure 6.9: Outdoor flight path through the ETHZ's hospital area.

is a subset of the following map. The result of the optimization procedure for each map is the position which assures optimal coverage of the area with the given team. This optimal positions are used as an input to the new map which is produced by the aerial robot which performs the mapping procedure. An aerial robot has flew over the Birmensdorf area and based on this flight eight successive maps of different sizes were produced and used as an input to the CAO algorithm. In Table 6.2.2 we present the performance of a team of four robots for the eight successive maps, in term of coverage percentage. The final map is similar to the one presented in Fig. 6.12. In all cases the proposed framework provides satisfactory results in terms of coverage percentage.

Table 6.1: Coverage percentage for different initial configurations in the Birmensdorf area.

(% of Coverage)			
Test Case	1	2	3
Initial Configuration	44.49	40.49	56.81
Final Configuration	98.55	99.52	99.56

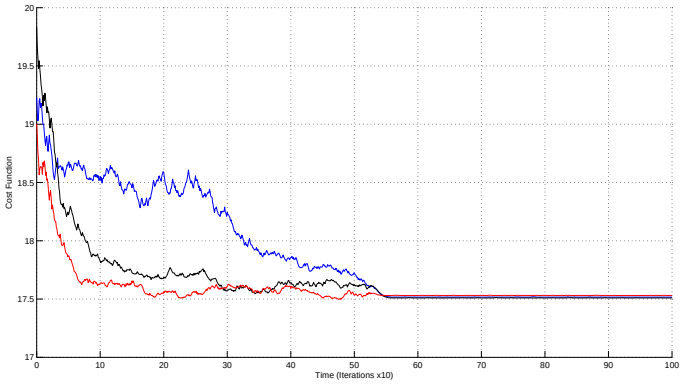


Figure 6.10: Comparative cost functions for different initial robot team configurations in Birmensdorf area.

6.2.3 Indoor area

Some simulations have been carried out also for indoor environments. ETHZ-CVG provided 3D maps which were produced by real flight data recorded by the prototype flying robots. These maps were processed and modified based on feedback from CERTH and INRIA in order to be compatible with the proposed methodologies. CERTH updated the software in which the optimization framework is implemented based on feedback from the sFly partners. The graphical user interface of the optimization framework is presented in Fig. 6.16.

6.3 Experimental results

We present here the results obtained during the final demonstration of the project. For the definition of the final scenario, we went back to the original vision of the sFly project and

Table 6.2: Incremental scenario in the Birmensdorf area.

Test Case	1	2	3	4	5	6	7	8
Initial % of coverage	69.83	85.37	63.82	65.57	49.94	75.32	74.2	81.21
Final % of coverage	94.5	98.01	95.44	95.32	72.56	79.56	76.72	90.5
% of the final map	5.46	6.55	9.63	16.86	59.98	70.23	81.8	100

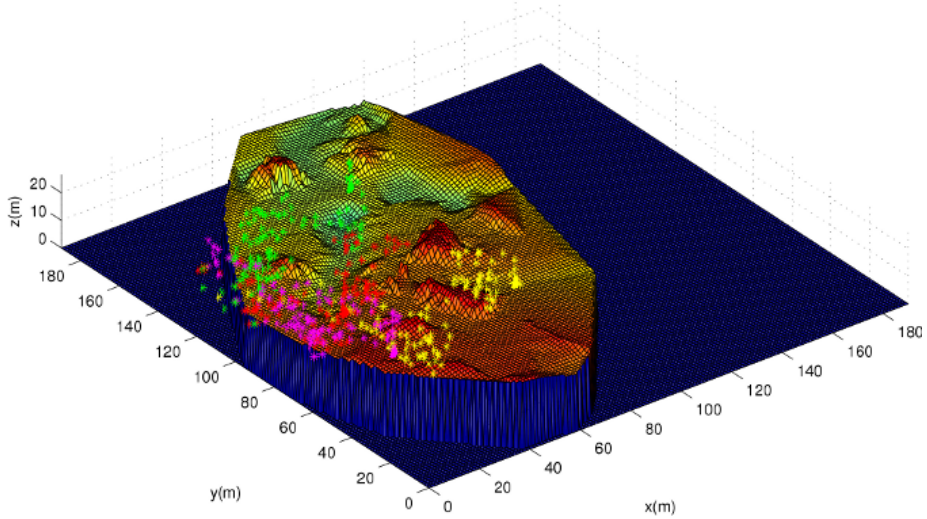


Figure 6.11: 3D Path followed by a robot team in a coverage scenario in Birmensdorf area.

the potential fields of application described therein. The biggest fundamental challenges thus consist of:

- robust monocular vision and inertial based autonomous hovering, navigation, and mapping with micro aerial vehicles in an unstructured environment,
- optimal surveillance coverage, and
- target localization with Received Signal Strength Indicator (RSSI) measurements.

The idea is to merge the solutions of all these challenging research problems in a single, combined search and surveillance mission. The underlying story is the support for the searching of a victim in a large scale environment. The victim is equipped with a transmitter badge for its identification and localization. The mission consists of first creating a common global map of the working area with three helicopters, then engaging positions for an optimal surveillance coverage of the area, and finally detecting the transmitter position. For safety and logistical reasons, the intended demonstration site must be a realistic and unpopulated outdoor area. The firefighter training area in Zurich has been selected as a perfect site for our purposes (see Fig. 6.17).

In this case, the helicopters used for the mission are hexacopters with a dual core processor, always developed by Ascending Technologies (Fig. 6.18). The localization process by RSSI measurements has been developed by CSEM.

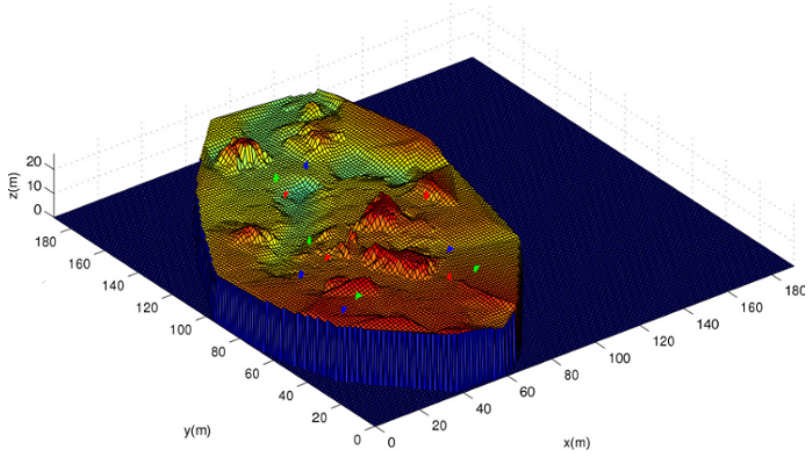


Figure 6.12: Final configurations of three robot teams starting from different initial positions for the Birmensdorf area.

6.3.1 Demonstration Scenario

The outline of the mission is the following:

- Preparation: place a transmitter node at an arbitrary position in the area.
- Take off with three MAVs from the same spot and explore the area in a manually set pattern with potential overlap in the scenes viewed by each helicopter. Stabilization and navigation is based on local visual mapping/visual odometry.
- Images are streamed to the ground station for off-line mapping after landing. The ground station merges the three local maps and generates a consistent global map, which is uploaded on each helicopter.
- Compute optimal coverage positions.
- Command the helicopters to anchor positions for transmitter node detection.
- Locate the transmitter node in the area by combining the RSSI measurements made by the MAVs.
- Hover at optimal coverage positions for live surveillance of the hypothetical rescue operation.
- Return to safe landing spots.

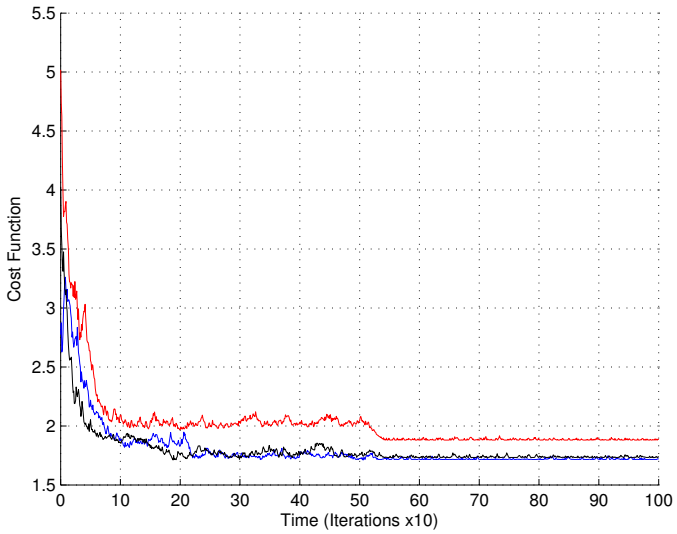


Figure 6.13: Comparative cost functions for different initial robot team configurations in ETHZ's hospital area.

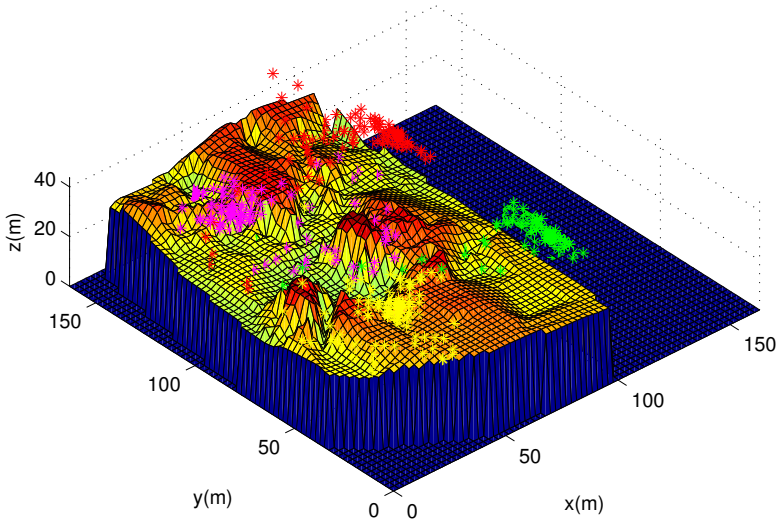


Figure 6.14: 3D Path followed by a robot team in a coverage scenario in the ETHZ's hospital area.

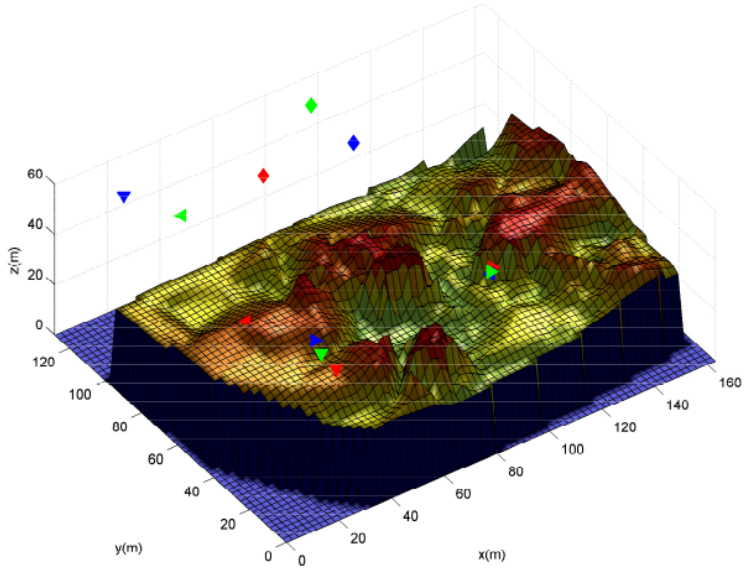


Figure 6.15: Final configurations of three robot teams starting from different initial positions for the ETHZ's hospital area.

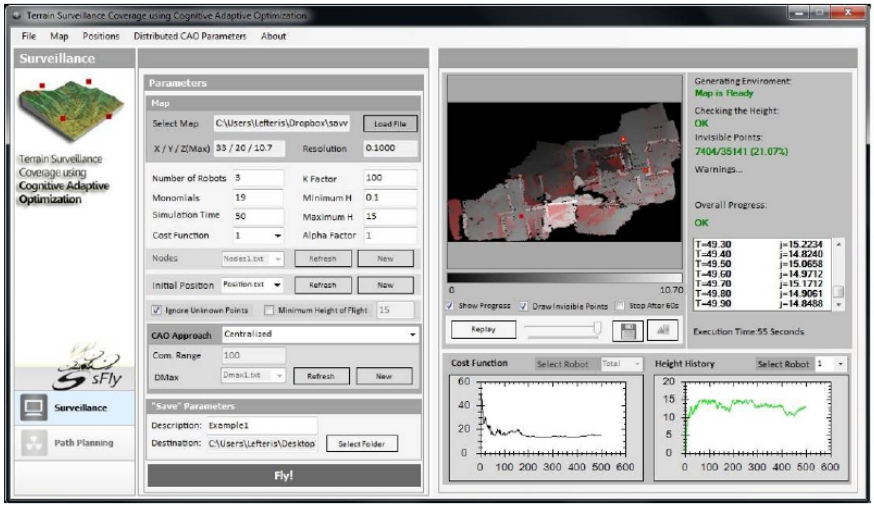


Figure 6.16: Graphical user interface of the CAO algorithm for a simulation in an indoor environment.

Figure 6.17: The firefighter training area in Zurich: the selected site for the final demo of the sFly project.

The node represent a victim in a workplace or factory scenario. Having the team of helicopters surveying the mission and tracking the victim from above allows for an efficient command execution of the entire operation from a centralized position. The absolute position as well as the close environment of the workers is known without any additional effort from the individual ground team members. The decision to perform take-off with the three helicopters from the same spot guarantees immediate overlap of the local maps, and thus eases the initialization of the global map. It leads to immediate mutual knowledge of the absolute as well as the relative positions of the vehicles, and practically enables the coordinated flight following the exploration pattern without making the demonstration less impressive or any of the subtasks simpler. This demonstration integrates the output of all the partners, and finally show the cooperative execution of a challenging multi-robot mission incorporating the major vision behind the sFly project.

During the demonstration, for practical reasons, after the exploration and mapping step, the three helicopters landed. Then, the map was reconstructed and the optimal coverage po-

Figure 6.18: During the final project demonstration three helicopters are called to map and cover an outdoor environment.

sitions calculated but only one helicopter took off again and autonomously flied through the optimal positions for the victim localization. At each of these positions it hovered for eight second. In Fig. 6.20, a screen-shot of the interface developed by CERTH is shown. This interface allows the user to easily set all the parameters of the system and visualize at the same moment a 2D view of the environment with the robots positions, the behavior of the cost function, the height of flight of the helicopters and the total covered surface. Finally, Fig. 6.21 shows a 3D view of the map of the environment with the final optimal positions.

6.4 Conclusions

In this chapter we presented the main application of our work, which was possible fusing the contributions of all the partners of the sFly project. Initially, a two-step procedure to align a swarm of flying vehicles to perform surveillance coverage has been presented and formally analyzed. A state-of-the-art visual-SLAM algorithm tracks the pose of the camera while, simultaneously, building an incremental map of the surrounding environment, autonomously, given that sufficient (arbitrary) visual features are available. In unprepared outdoor environments, such a requirement of having sufficient features is generally fulfilled. The reconstructed mesh map of the environment is used as the input to the second part of the procedure where the CAO methodology is used to maximize the area monitored by a team of aerial robots. Using this procedure, several simulations both in outdoor regions near Zurich and indoor areas has been carried out. Finally, during the final project demonstration, a real experimentation using

(a) (b)

Figure 6.19: The three helicopters flying during the final demonstration.

three micro helicopters has been accomplished. The goal of the task was to localize a victim in a complex outdoor environment, reproducing a typical search and rescue mission. After a first step of exploration and mapping following manually defined trajectories, but with a vision-based flight, the map has been used as the input of the CAO algorithm to calculate the final optimal coverage positions. Then, a helicopter hovered on these positions and localized a victim, positioned at the beginning of the task in an arbitrary point of the environment. This demonstration had the goal to combine the work of all the partners of the project and simulate a realistic rescue mission in a GPS-denied environment.

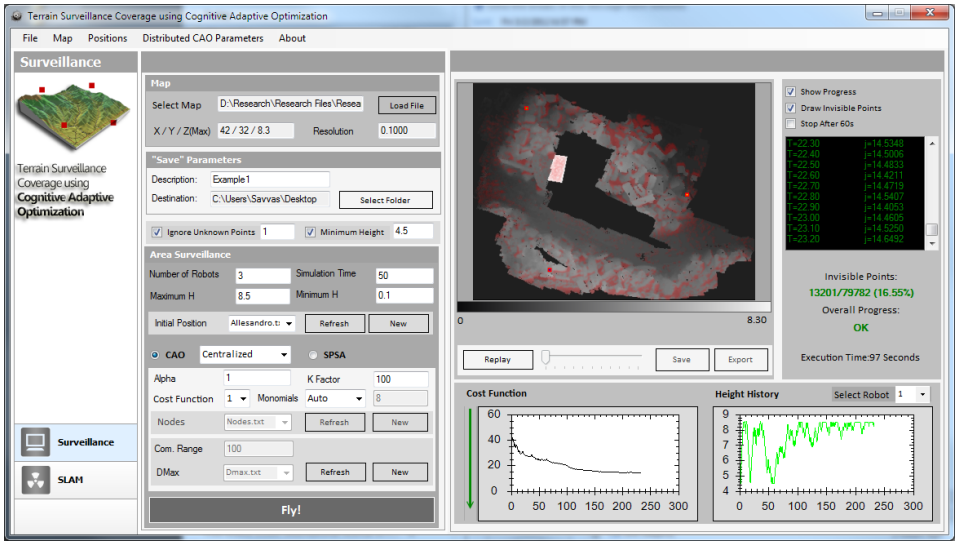


Figure 6.20: Screen-shot of the interface for the calculation of the optimal coverage positions.

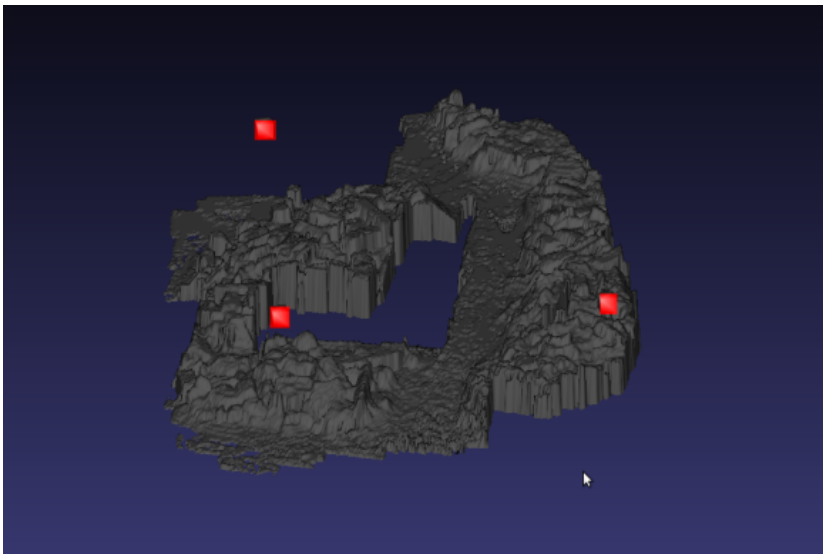


Figure 6.21: The map obtained by the three helicopters and the final optimal positions in a 3D view.

Figure 6.22: The localization of the victim is the final goal of the task.

Chapter 7

Navigating between People

As already sentenced, the CAO algorithm may be very useful for many different guidance problems in complex and dynamic environments, where an adaptive, fast, sensor-based approach is of crucial importance. To demonstrate that, in this chapter we consider a problem completely different from the cooperative surveillance coverage and we show how the same algorithm can be used to obtain a solution also in this case. The problem chosen for this purpose is: safely moving a robot in an unknown and complex environment where people are moving and interacting. The robot must navigate respecting humans' comfort. A typical scenario with a wheel chair moving between humans is shown in Fig. 7. To obtain good results in such environments, a prediction on humans' movement is also crucial. To solve all the aforementioned problems we introduce a suitable cost function. The results of this chapter have been presented in [97].

7.1 Introduction

Robots navigating close to humans or involved in interaction tasks with humans must assure not only safe but understandable behavior in order to prevent discomfort in people. Recently, several possible solutions to this problem have been proposed [107, 56, 68, 98]. Our work is placed in this framework: we are interested in safely lead a robot in an unknown and complex environment, where people are moving and interacting, respecting the humans' comfort. The first step is to understand how humans manage the space around them while navigating and how their decisions affect the comfort of others. Many psychological theories have been proposed to explain the relation between distance, visual behaviors and comfort in humans (see [3] and references therein). Intuitively people will become uncomfortable if they are approached at a distance that is judged to be too close: the greater invasion/intrusion the

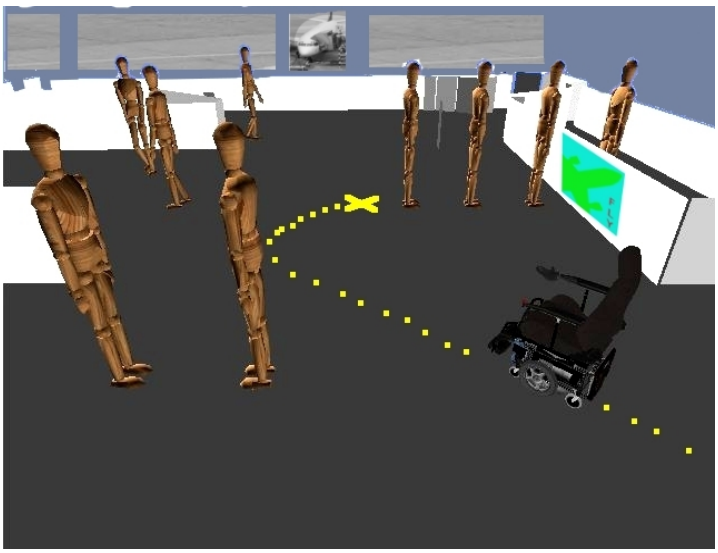


Figure 7.1: A typical scenario of navigation between humans. A wheelchair is moving in an area where people are chatting and interacting.

more discomfort or arousal is experienced by the person. This simple idea was formalized introducing the concept of personal space, firstly proposed by Hall [42], which characterizes the space around a human being in terms of comfort to social activity. In casual conversations, people claim an amount of space related to that activity. This space is respected by other people and only participants have permitted access to it, therefore the intrusion of a stranger causes discomfort [53]. It can be assumed that people will engage in proxemic behavior with robots in much the same way that they interact with other people [114]. For example in [21], participants evaluated the direct frontal approach as least comfortable for a bring object task by finding robots motion threatening and aggressive.

In this chapter we formulate the problem of social robot navigation as an optimization problem where the objective function includes, in addition to the distance to goal, information about comfort of present humans. Using the CAO algorithm to generate the trajectory, we can also obtain an indirect prediction on the people movement, which is a very crucial point to get good results for a similar task.

7.1.1 Related Work

A proposal of human aware navigation was presented in [107], where it is implemented a motion planner which takes explicitly into account its human partners. The authors introduced criteria based both on the control of the distance between the robot and the human, and on the control of the position of the robot within the human's field of view. The criterion of

visibility was simply based on the idea that when the robot is in the field of view of a person, the comfort increases. Our assumption is in some way the opposite of the last criterion: the field of view shows the point of interest of a person then, if the robot enters it, the activity of the person will be interrupted decreasing the comfort function. The work presented in [68] proposed six harmonious rules that a single robot should obey in order to achieve not only a safe but also a least disturbance motion in a human-robot environment. Based on these rules, a practical reactive navigation algorithm was developed. It is considered the fact that both humans and robots have their sensitive zones, depending either on their security regions or on psychological feeling of humans. Personal space, o-space and their relation to comfort were addressed in [98], where a risk based navigation was extended to include risk due to discomfort. The method is based on a probabilistic version of Rapidly-exploring Random Trees. Human's movement is supposed to be known by learning of typical trajectories in a particular environment. Distance to goal, risk of collision and risk due to comfort are used as heuristics to take decisions but optimality of chosen paths is not guaranteed. The work presented in [72] investigates the notion of comfort and proposes some ways to use it in robotics domain. The goal is to identify the salient features in the environment that affect the comfort level. In [56] a generalized framework for representing social conventions as components of a constrained optimization problem was presented and it was used for path planning and navigation. Social conventions were modeled as costs for the A* planner with constraints like shortest distance, personal space and pass on the right. In contrast with the previous works, we can take advantage of information about past people positions to obtain a humans' movement prediction. This fundamental advantage is based on the possibility to work with an unknown objective function.

7.2 Proposed Solution

In this section we formulate the particular problem we want to solve and we show how the proposed optimization algorithm can be applied in practice to the problem studied in this chapter. Furthermore, we discuss how it is possible to include a prediction of humans' motion by using the CAO algorithm.

Our intent is to safely move a robot in a complex and unknown environment respecting the comfort of the people moving in. Let $x_0^{(R)}$ be the robot start position and let $x^{(G)}$ be the goal position. Our intent is to move the robot from $x_0^{(R)}$ to $x^{(G)}$ minimizing the discomfort of humans located at positions $\{p^{(i)}\}$. The discomfort function has two components, one for the invasion of Personal Space ($dis(PS)$) and the other for invasion of Information Process Space

($dis(IPS)$), both of them explained later in this section. To fulfill both the tasks of reaching the goal and respecting the people, we define the optimization function in the following way:

$$J = \lambda * (dis(PS) + dis(IPS)) + D(x^{(G)}) \tag{7.1}$$

where λ is a constant parameter and $D(x^{(G)})$ is a function depending on the distance to the goal. In our case it is the Euclidean distance.

The difference with respect to the application of the algorithm, provided in previous chapters, is that now the cost function depends on both active variables (the robot's position $x^{(R)}$) and passive variables (humans' positions $\{p^{(i)}\}$). This means that now the cost function can be expressed in the form:

$$J = J(x^{(R)}; \{p^{(i)}\}) \tag{7.2}$$

and only the controllable components $x^{(R)}$ are perturbed to generate the candidate new positions.

7.2.1 Discomfort model

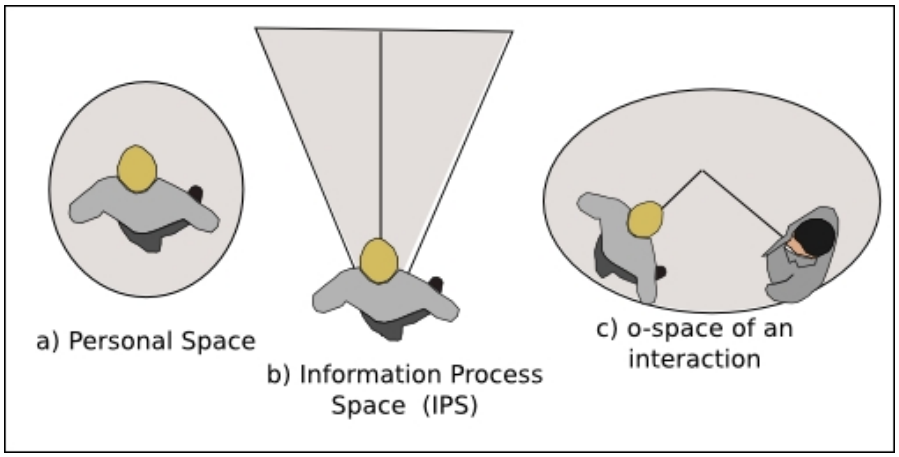


Figure 7.2: We consider as discomfort the invasion made to humans' space by the robot, specifically, a) Personal Space b) Information Process Space or c) o-space.

Since comfort is a subjective notion it is clear that it cannot be measured directly by any sensor, however some studies have been developed to explain how distance and visual behavior affect comfort in humans (see [4, 3]). Some other works have studied the visual behavior of pedestrians when navigating: for example in [57] authors explored the size and the shape of Information Process Space (IPS), in which a pedestrian takes account of other pedestrians and obstacles for calculating next moves and where psychological comfort is evaluated (this space

can be related to visual field). We built our comfort model based on these works. We consider as discomfort the invasion made to humans' space, specifically personal space [42], o-space [53] and Information Process Space [3], by the robot. A representation of these spaces can be observed in Fig. 7.2. We assume that the discomfort will be higher in the spaces previously mentioned and we propose a function that approximates them. The function to represent IPS is inspired on the representation of the Doppler effect which establishes that the perception in the frequency of a sound varies with the movement of source and observer. The source of sound is a pedestrian that moves with a constant velocity and all the other points are observers which do not move. Then the equation is:

$$f' = \frac{c}{c - v_s \cos\theta_s} f \,, \tag{7.3}$$

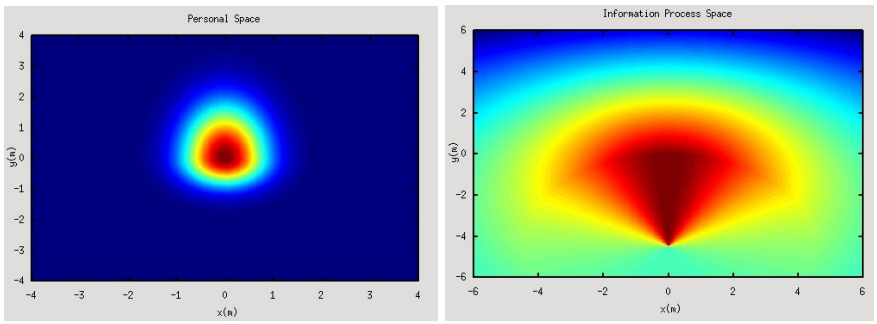
where f is the frequency emitted by the source, f' is the frequency perceived by the observer, c is the velocity of sound, v_s is the velocity of the source and θ_s is the angle between the direction of the source and the direction of the vector linking observer and source. Numerical values for the parameters in eq. (7.3) have been determined empirically to best adjust to the results for IPS in [57], and they are $c = 3.43$, $v_s = 3.0$; f is determined in function of distance as stated in next equation:

$$f = \begin{cases} 1 & if \quad d < d_e \\ 1 - \left(\frac{d - d_e}{d_l}\right) & if \quad d_e \leq d \leq d_e + d_l \\ 0 & if \quad d > d_e + d_l \end{cases} \tag{7.4}$$

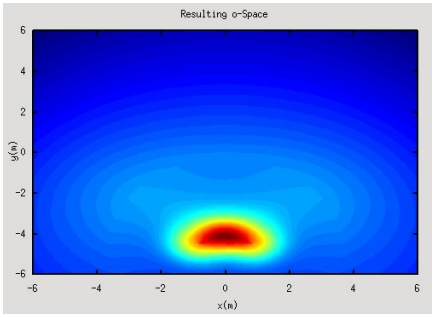
where d is the distance from the human's position, d_e is the main ratio of effect of the IPS and d_l is the ratio where the IPS loses its effect. In our current implementation $d_e = 4.5$ and $d_l = 4.5$.

When two people are interacting the o-space is created by the intersection of the two IPS, as we can see in the case presented in Fig. 7.3 (c). Finally, we use a Gaussian function centered on the pedestrian position to represent the Personal Space; the front is wider than the back as presented in [56].

Using these equations we can get the next graphics for the models: the first one is the Personal Space for a pedestrian walking in the direction of y-axis, the second one the IPS for the same case and the third one shows the resulting o-space for two pedestrians in conversation. The robot must avoid the red regions while navigating.



(a) (b)



(c)

Figure 7.3: Models implemented to represent discomfort in humans' spaces: (a) Personal Space of a human in (0,0) and orientation of 90 degrees. (b) IPS for a human in (0,-4.5) and orientation of 90 degrees. (c) O-space for two humans in positions (-0.85,-4.5) and (0.85,-4.5) and orientations of 30 and 150 degrees, respectively. Higher discomfort in darker red, lower in lighter blue.

7.2.2 Movement Prediction

As already stated, our intent is to consider a dynamic environment where the people $\{p^{(i)}\}$ are moving. The objective function is then time-dependent and in general it will be different for each time step:

$$J_t = J(x^{(R)}; \{p_t^{(i)}\}) \,. \tag{7.5}$$

In this case, in order to solve the optimization problem, i.e. finding the optimal next robot position, the result can be considerably improved if we consider the function J_{t+1} instead of J_t, where:

$$J_{t+1} = J(x^{(R)}; \{p_{t+1}^{(i)}\}) \,. \tag{7.6}$$

This function is obviously unknown at time t but it could be approximated if a prediction model is available. Indeed, we can express the positions $\{p_{t+1}^{(i)}\}$ by means of a limited set of q

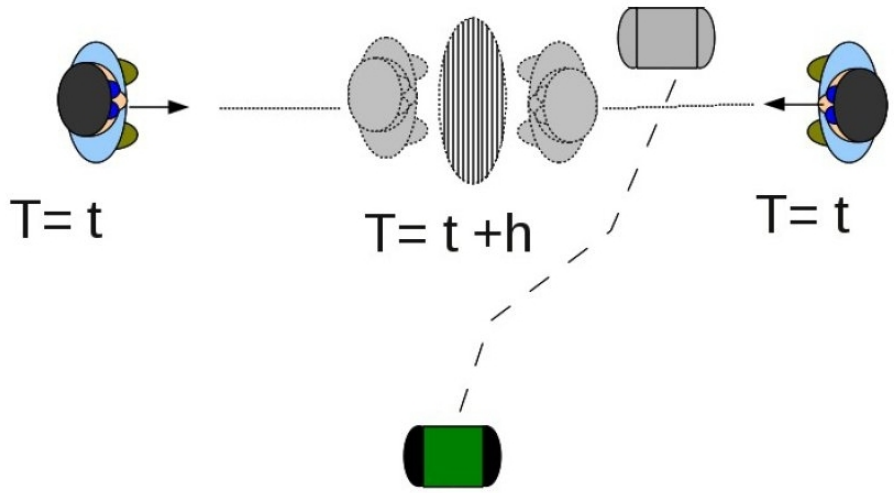


Figure 7.4: An example of prediction: the robot anticipates humans' movements and avoids them.

past configurations

$$\{p_{t+1}^{(i)}\} = g(\{p_t^{(i)}\}, ..., \{p_{t-q}^{(i)}\}) \tag{7.7}$$

where the new function g represents the prediction model. In our case we do not assume any particular model and the function g is to consider completely unknown. Hence also the function

$$J_{t+1} = J(x^{(R)}; g(\{p_t^{(i)}\}, ..., \{p_{t-q}^{(i)}\})) \tag{7.8}$$

is now unknown. The strategy to approach the problem is not to explicitly predict the humans' movement but try to directly approximate the cost function (7.8) using its available past values. To do this in practice, we construct at each time step an approximator $\hat{J}_t$, like in (3.11), of the unknown function J_{t+1} using the last $m > q$ numerical values of J_t such that:

$$\hat{J}_t(x_t^{(R)}; \{p_{t-1}^{(i)}\}, ..., \{p_{t-q-1}^{(i)}\}) \approx J(x_t^{(R)}; \{p_t^{(i)}\})\,. \tag{7.9}$$

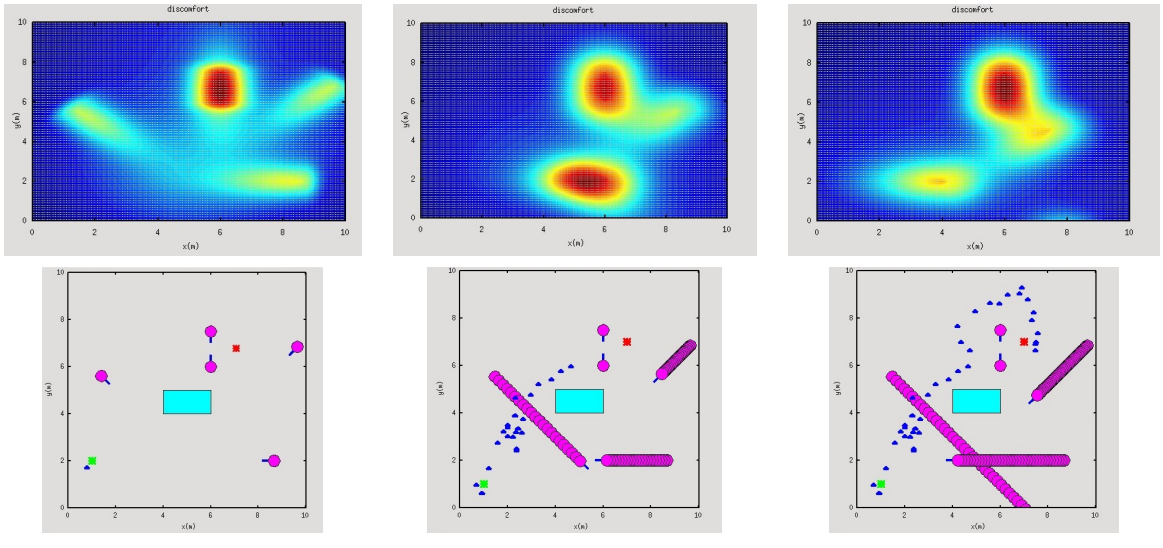
In this way, using the last available set of humans' positions, we have an indirect approximation of the humans' movement prediction and we obtain

$$\hat{J}_t(x^{(R)}; \{p_t^{(i)}\}, ..., \{p_{t-q}^{(i)}\}) \approx J_{t+1} \tag{7.10}$$

i.e., the function we want to optimize.

7.3 Performance Evaluation

In this section several scenarios are presented to show the execution of our algorithm in simulation. The first scenario is shown in Fig. 7.5: in this case five humans are present, three of them are moving and two interacting. The robot starts at (1,1) and reaches its goal while avoiding people and o-space of interaction. In Fig. 7.6 four different and more complex scenarios are presented. In (a) a robot has to pass through a corridor while two humans are chatting in the middle. It is possible to see how the robot is able to understand the interaction and to avoid them without disturbing. We can notice how the method evaluates many points that fall in the shortest path but finally can found a more comfortable way. In Fig. 7.6(b), the robot start position is aligned with the goal position but as one people is looking to the walls the chosen path guides the robot toward the middle of the corridor and then to the goal. We can remark that in this case, since the two people are not interacting, the robot can pass between them without trouble. A representation of a room with people inside is exhibited in Fig. 7.6(c). Here the chosen path does not interrupt any human. Last example is shown in Fig. 7.6(d), where the robot respects o-space of the group and p-space of humans. Note that in every simulation the presence of obstacles does not create any problem to the robot navigation. Additionally the proposed algorithm, due to the random generation of next state configuration, is able to overcome many of typical local minima generated by obstacle avoidance problems.



(a) (b) (c)

Figure 7.5: Simulation of the robot navigating in an environment populated by people at three different times. Three humans walking and two in conversation. The discomfort function is shown on the top. People are represented by circles, robot's positions by small triangles, in green and red initial and goal position respectively.

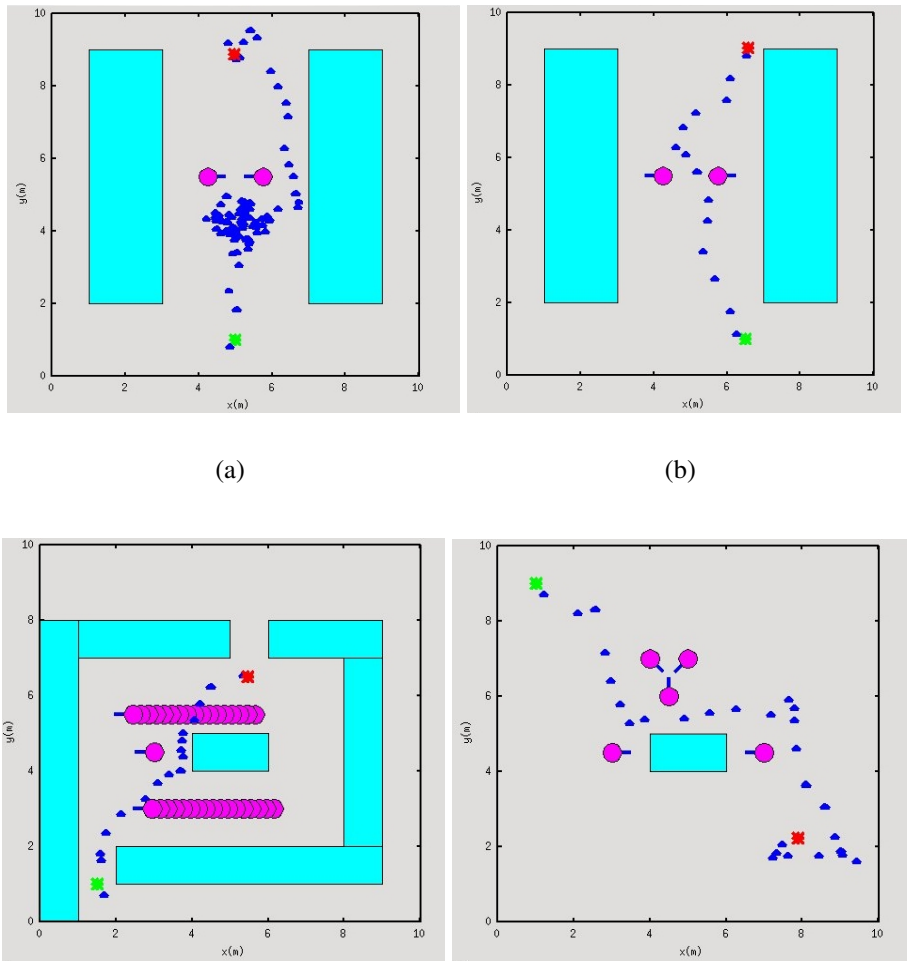
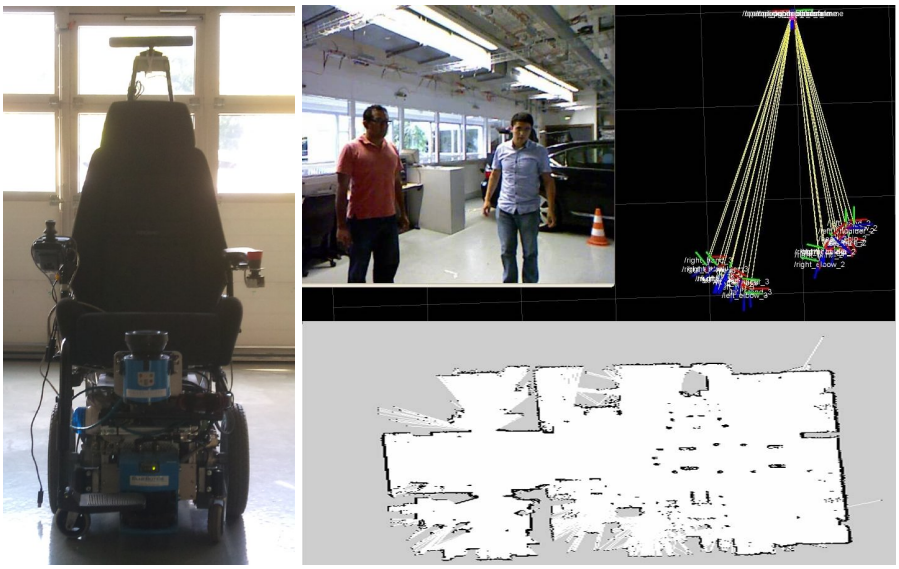


Figure 7.6: More simulations with different scenarios. Start positions are in green, goal positions in red. In (a) the robot decides to take a path that minimizes discomfort of interacting humans. In (b) a similar configuration but now humans are not interacting. In (c) and (d), a pair of different complex scenarios where the robot's trajectories respect people comfort.

7.3.1 Experimental platform

Due to the promising results achieved by simulations, the proposed approach is being implemented on our experimental platform, an automated wheelchair (Fig. 7.7(a)) equipped with two Sick lasers and a Microsoft Kinect, running ROS (Robotic Operating System) for achieving semi-autonomously mobility actions commanded by the wheelchair's user. Laser permits us to build a map of the environment, like shown on the bottom of Fig. 7.7(b). Data coming from the Kinect will allow us to have position and orientation of pedestrians in the scene.



(a) (b)

Figure 7.7: Experimental platform: in (a) the wheelchair, on the top of (b) the data provided by the kinect, on the bottom the final map.

7.4 Conclusions

In this chapter we wanted to show how the stochastic optimization algorithm exploited in this thesis can be applied for many and very different robotic problems. Indeed, as already stated, the many advantages of this approach make it very suitable for applications in mobile robotics. Particularly in presence of complex, dynamically and uncertain environments. The task considered in this chapter was to navigate a robot, from an initial position to a predefined goal, respecting the people comfort and avoiding the obstacles. We assumed the robot has not an a priori knowledge about the environment, so the obstacles are detected during the navigation. The results obtained in this chapter are a strong motivation to continue the research and to implement the method in a real dynamic environment using the wheelchair previously described.

As new sensors are becoming available to measure social signals we can take advantage of them in our framework. In particular, Sociometric Badges ([83]) will be directly used by our method to get high level descriptions of human behavior, mainly in the case of face-to-face interaction and physical proximity.

Conclusions

In this thesis, developed in the framework of the European project sFly, we have presented a new approach based on an adaptive and stochastic algorithm, to achieve cooperative tasks. The cooperative coverage problem has been studied with a final application for surveillance of a complex outdoor region by using a swarm of Micro Aerial Vehicles (MAVs). The adopted optimization algorithm (the CAO algorithm), recently proposed by Kosmatopoulos in [63, 60], has been suitably modified to be applied to the considered problem. In particular this algorithm, applied to a robotic problem for the first time in this thesis, has been extended to include a strategy to take into account constraints and it has been proven that this new capability does not affect the convergence properties. This method is able to deal with optimization problems where the objective function is unknown in its analytical expression but numerical values are available for measurements given a state configuration. As a results, it is very useful for the problem we approached, where the terrain to cover can be unknown and/or too complex to be described in an analytical way. More precisely, the proposed algorithm has the following key advantages:

- it does not require any a priori knowledge on the environment;
- it works in any given environment, without the necessity to make any kind of assumption about its topology;
- it can incorporate any kind of constraints;
- its complexity is low allowing real time implementations;
- it requires low weight and low cost sensors, which makes it ideal for aerial robot applications;
- it builds autonomously the metric map required for the optimization procedure.

Furthermore, we have proposed a distributed version of this algorithm for a particular coverage criterion: maximizing the surface of the terrain that the team is able to see. This approach

is closer to real world applications since it does not suffer from the well known disadvantages of a centralized method. In addition, a decentralized approach will allow us to include communications constraints.

The first step to validate our method was to perform simulations in 2D area, where the problem is easier to solve and the results are more understandable. For this first case, we also proposed a completely different approach, based on the artificial potential field method and the Lloyd algorithm, to solve the Voronoi coverage problem in an unknown non-convex region. Then, we continued our investigation by extending the algorithm to make it able to deal with 3D environments. The first simulations were carried out using complex simulated environments and considering two different possible coverage criteria. Motivated by the good results obtained in this phase, we have used real data provided by a helicopter. These data have been collected in complex, outdoor regions near Zurich and we used them to reconstruct a map of the environment, which served as an input for our algorithm. Particular incremental scenarios are also tested, where the coverage is achieved simultaneously with the mapping. The final step of our work has been the implementation of the algorithm on a real swarm of MAVs to perform a surveillance coverage mission. A coverage mission in an outdoor area in Zurich has been carried out with the aim of localize a victim positioned in an arbitrary point of the environment.

Finally, we have approached also a different problem of navigation. In this case the goal was safely navigating a robot in an unknown and complex environment where people are moving and interacting. The aim of the robot was respecting the comfort of the people. Also for this application the results were completely satisfactory and an implementation on an experimental platform is expected.

We expect that many important tasks in mobile robotics can be approached by CAO-based algorithms, for example: coordinated exploration, optimal target tracking, multi-robot localization, and so on. This is basically due to the fact that the CAO approach does not require an a priori knowledge of the environment and it demands low computational resources. Both these issues are fundamental in mobile robotics.

Future work

A first future work we are carrying out is a detailed comparison between the proposed method and other stochastic optimization method. In particular, we are testing the SPSA algorithm on the coverage problem studied in this thesis. The SPSA algorithm has been described in chapter 2 and it is one of the most studied and exploited stochastic optimization algorithms. Prelimi-

nary results seem to show that the CAO algorithm, despite it requires less robots movements, can converge faster to a local optimum.

All the results of this thesis proved that the advantages of the proposed stochastic methodology make it suitable for real implementations and the results obtained give us the motivation to adopt the CAO also in other frameworks. We are now extending the method to approach problems of active target tracking and simultaneous localization and mapping (SLAM). Preliminary results have already obtained. We believe they are promising and justify to continue the research on this topic.

Furthermore, we are interested in enhancing the proposed methodology in order to be able to deal with cases where the team converges to a configuration that fails to cover the entire terrain. In other words, we are trying to improve the ability of the algorithm to overcome local minima. It is worth noticing, that although in the 3D case treated here we have never encountered very evident local minima, there is always the possibility (due to the local convergence properties of the algorithm) for the methodology to fail to cover the overall area. As a matter of fact, such cases have been encountered in the 2D version of the algorithm and an extension/enhancement of the proposed methodology is required to deal with such cases. Another important example is when the team capabilities are not sufficient to assure the surveillance of the entire environment. In this case a static deployment is not appropriate and a dynamical coverage strategy is necessary.

We are also interested into formulating the new 3D coverage proposed criterion, introduced in Section 5.4, in a distributed manner by using different cost functions for the robots in the team. Moreover, the results we obtained with the distributed algorithm for the visibility problem were more satisfactory than we expected and they deserve a deeper analysis.

Finally, about the last application of the CAO approach presented in Chapter 7, in addition to the implementation the algorithm on the wheelchair for a real experimentation, we want to compare our results with other obtained with already existing methods. Moreover, an accurate statistical analysis to validate the expected movement prediction is an important step.

List of Publications

Journals

1. A. Renzaglia, L. Doitsidis, A. Martinelli and E.B. Kosmatopoulos, ”Multi-Robot 3D Coverage of Unknown Areas,“ *The International Journal of Robotics Research,* 2012.

2. L. Doitsidis, S. Weiss, A. Renzaglia, M. Achtelik, E.B. Kosmatopoulos, R. Siegwart and D. Scaramuzza, ”Optimal Surveillance Coverage for Teams of Micro Aerial Vehicles in GPS-Denied Environments using Onboard Vision,“ *Autonomous Robots,* 2012.

Book Chapters

3. F. Conte, A. Cristofaro, A. Renzaglia and A. Martinelli, ”Cooperative Localization and SLAM Based on the Extended Information Filter,“ Multi-Robot Systems, Trends and Development, Toshiyuki Yasuda (Ed.), InTech, 2011.

Peer-Reviewed Proceedings

4. A. Renzaglia and A. Martinelli, ”Distributed Coverage Control for a Multi-Robot Team in a Non-Convex Environment,“ in *IEEE IROS09 3rd Workshop on Planning, Perception and Navigation for Intelligent Vehicles,* St Louis, MO, USA, pp. 76-81, 2009.

5. A. Renzaglia and A. Martinelli, ”Potential Field based Approach for Coordinate Exploration with a Multi-Robot Team,“ in *8th IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR),* Bremen, Germany, 2010.

6. A. Renzaglia, L. Doitsidis, A. Martinelli and E.B. Kosmatopoulos, ”Cognitive-based Adaptive Control for Cooperative Multi-Robot Coverage,“ in *IEEE International Conference on Robotics and Intelligent System (IROS),* Taipei, Taiwan, pp. 3314-3320, 2010.

7. A. Cristofaro, A. Renzaglia and A. Martinelli, ”Distributed Information Filters for MAV Cooperative Localization,“ in *10th International Symposium on Distributed Autonomous Robotics Systems (DARS),* Lausanne, Switzerland, 2010.

8. A. Renzaglia, L. Doitsidis, A. Martinelli and E.B. Kosmatopoulos, ”Adaptive-based, Scalable Design for Autonomous Multi-Robot Surveillance,“ in *49th IEEE Conference on Decision and Control (CDC),* Atlanta, GA, USA, pp. 4618-4624, 2010.

9. A. Renzaglia, L. Doitsidis, A. Martinelli and E.B. Kosmatopoulos, ”Adaptive-based Distributed Cooperative Multi-Robot Coverage,“ in *IEEE American Control Conference (ACC),* San Francisco, CA, USA, pp. 468-473, 2011.

10. L. Doitsidis, A. Renzaglia, S. Weiss, E.B. Kosmatopoulos, D. Scaramuzza and R. Siegwart, ”3D Surveillance Coverage Using Maps Extracted by a Monocular SLAM Algorithm,“ in *IEEE International Conference on Robotics and Intelligent System (IROS),* San Francisco, CA, USA, pp. 1661-1667, 2011.

11. A. Martinelli, C. Troiani and A. Renzaglia, ”Vision-Aided Inertial Navigation: Closed-Form Determination of Absolute Scale, Speed and Attitude,“ in *IEEE International Conference on Robotics and Intelligent System (IROS),* San Francisco, CA, USA, pp. 2460-2465, 2011.

12. A. Renzaglia, L. Doitsidis, A. Martinelli and E.B. Kosmatopoulos, ”Multi-Robot 3D Coverage of Unknown Terrains,“ in *50th IEEE Conference on Decision and Control (CDC),* Orlando, FL, USA, pp. 2046-2051, 2011.

13. J. Rios-Martinez, A. Renzaglia, A. Spalanzani, A. Martinelli and C. Laugier, ”Navigating Between People: a Stochastic Optimization Approach,“ in *IEEE International Conference on Robotics and Automation (ICRA),* St. Paul, MN, USA, 2012.

Appendix A

Resumé en Français

A.1 Introduction

L'utilisation déquipes de véhicules aériens sans pilote (UAVs) a pris de l'ampleur dans les dernières années. Cela est dû aux avantages que les robots volants peuvent offrir par rapport à l'utilisation de robots terrestres pour la même tâche. La capacité de voler permet facilement d'éviter les obstacles sur le terrain et d'avoir une excellente vue d'oiseau. En outre, il est possible d'accéder à des environnements où des humains ou d'autres véhicules ne peuvent pas accéder. Par conséquent, les robots volants sont les héritiers logiques des robots mobiles terrestres. Si en plus, ils sont réalisé à petite échelle, ils peuvent aussi être utilisés dans des environnements étroits extérieurs et intérieurs et le risque pour l'environnement et les gens qui y vivent n'est que limité. Des équipes de Micro-Véhicules Aériens (MAV) peuvent être utilisés dans une variété de missions très importantes, y compris:

- Surveillance de bâtiments et de grandes espaces extérieurs et intérieurs: au lieu de caméras fixes de sécurité, les micro-hélicoptères permettraient une grille re-configurable de caméras de surveillance et en cas de besoin ils peuvent établir une dans des endroits où les caméras de sécurité n'existent pas.

- Missions de sauvetage: des robots aériens capables de voler dans des quartiers fermés et des bâtiments effondrés pourraient rapidement et systématiquement chercher à localiser les victimes d'un accident ou d'une catastrophe naturelle.

- La surveillance des zones dangereuses, des centrales chimiques et nucléaires: les micro-hélicoptères pourraient explorer des domaines dangereux pour l'homme, comme des zones de contamination chimique, biologique ou nucléaire.

Figure A.1: Scénarios typiques où la surveillance est une tâche fondamentale.

- Surveillance de l'environnement: les micro-robots volants sont un excellent outil pour la surveillance de l'environnement (qualité de l'air, feux de forêt, ...) comme unité individuelle, dans une équipe ou en relation avec un réseau de capteurs.
- Application de la loi dans la zone publique: un micro-hélicoptère pourrait fournir des images en temps réel pour aider la police au cours des missions de surveillance ou des opérations de recherche criminelle.

Dans toutes les tâches mentionnées ci-dessus le déploiement de ressources limitées (robots) pour optimiser la surveillance de la zone d'intérêt est la question clé. En outre, comme ces plates-formes deviennent de plus en plus abordables et robustes, l'utilisation d'équipes de véhicules aériens que, de manière coopérative et autonome explorent et couvrent une superficie affectée, devient une alternative viable. Afin d'exploiter les avantages de la mobilité des robots, des stratégies de détection actives doivent être déterminées pour coordonner le mouvement de groupes de robots.

Dans tous les systèmes multi-robots, une approche distribuée est souhaitable pour plusieurs raisons fondamentales. Les plus importantes sont l'échec de la gare centrale et des capacités de communication limitées. Dans un scénario très fréquent, chaque robot n'a pas de connaissances globales sur l'environnement ou sur le groupe dans son ensemble. Ainsi, le comportement global de l'équipe peut être considérée comme la somme des actions locales menées par ses membres, qui détectent leur environnement immédiat, communiquent avec leurs voisins, traitent l'information recueillie et se déplacent selon elle.

A.1.1 Contexte de la thèse

Le travail de cette thèse a été réalisé dans le cadre du projet européen sFly (www.sfly.org). L'objectif de ce projet est de développer plusieurs petits hélicoptères qui peuvent voler de façon autonome dans des environnements urbains et qui peuvent être utilisés pour aider les humains dans des tâches telles que le sauvetage et la surveillance. Les principales motivations

Figure A.2: Trois hexacopters utilisés pour le projet sFly.

de ce travail sont non seulement de réaliser des tâches impossibles pour une équipe humaine, mais aussi pour être en mesure de remplacer l'intervention humaine dans des scénarios très dangereux. Cela signifie que les hélicoptères doivent être en mesure de fonctionner dans des environnements complexes où les signaux GPS sont souvent ombragés, en collaboration et en complète autonomie. Cela implique un certain nombre de défis à tous les niveaux de la conception d'hélicoptères, de la perception, de l'actionnement, du contrôle, de la navigation et de l'alimentation qui n'ont pas encore été résolus. Les problématiques de la navigation basée sur la vision ont été abordées par l'Autonomous System Lab (ASL) et le Computer Vision and Geometry Group (CVG) à l'ETH de Zurich, en Suisse. Les hélicoptères ont été mis au point pour le projet par Ascending Technologies GMBH (ATG), Allemagne (voir Fig. A.2). Les questions de communication sans fil et de mesures de distance sont considérés par le CSEM, Neuchâtel, Suisse. Enfin, l'INRIA (Grenoble, France) et CERTH/TUC (Thessalonique/ La Canée, Gréce) ont été les partenaires chargés de la direction de navigation active pour les tâches de coopération, le sujet de cette thèse.

A.1.2 Couverture Coopérative

Le problème du déploiement d'une équipe de robots volants pour effectuer des missions de couverture de surveillance sur un terrain inconnu de morphologie complexe et non-convexe est considéré. Ce problème peut être exprimé comme un problème d'optimisation: étant donné une configuration d'équipe initiale arbitraire, trouver les positions finales des robots qui maximisent le degré de couverture et la façon de parvenir à une telle configuration. Dans cette thèse, nous supposons que les capacités de surveillance de l'équipe sont suffisantes pour atteindre un niveau satisfaisant de surveillance à partir d'une configuration statique. En d'autres

termes, nous ne considérons pas ici le cas d'un environnement trop grand pour être surveillé et qui nécessite des stratégies de surveillance dynamiques. Afin de quantifier le degré de couverture pour une mission de typique couverture deux différents critères principaux peuvent être identifiés:

(O1) la partie de surface qui est surveillée par l'équipe de robots doit être maximisée;

(O2) pour chaque point dans le terrain, le robot le plus proche doit être le plus proche possible de ce point.

Le premier objectif est le plus intuitif dans une tâche de surveillance: trouver les positions à partir desquelles il est possible de voir le plus possible, en ce qui concerne les capacités des capteurs de l'équipe. Dans la thèse, nous allons nous référer à ce problème comme le problème de visibilité.

Le deuxième objectif pourrait être nécessaire pour deux raisons pratiques: (a) d'une part, dans nombreuses applications de couverture, il y a la nécessité d'être en mesure d'intervenir aussi vite que possible dans l'un des points du terrain avec au moins un robot et (b) d'autre part, plus proche est le robot à un point sur le terrain mieux est, en général, sa capacité de détection de surveiller ce point. Nous nous référerons à ce problème comme le problème d'intervention.

Trouver les positions optimales pour l'équipe de robots n'est pas l'unique problème à résoudre. En effet, dans de nombreuses situations le problème d'optimisation doit être résolu en ligne, à partir de positions complètement arbitraires, sans, ou avec une partielle connaissance a priori sur l'environnement à surveiller. Donc dans ce cas, également la création des trajectoires en temps réel, respectant toutes les contraintes dynamiques et de l'environnement, constitue un autre défi.

Notre objectif est de développer une stratégie efficace et adaptative pour diriger les robots afin de maximiser la partie du terrain qui est visible, en gardant la distance entre chaque point sur le terrain et le membre le plus proche de l'équipe, le minimum possible. Un compromis entre ces deux objectifs doit être rempli, en prenant en compte des contraintes physiques et des limitations imposées par l'application particulière. Comme la morphologie du terrain est inconnue et peut être très complexe et non-convexe, des algorithmes standards ne sont pas applicables au problème particulier traité dans cette thèse. Pour surmonter cette difficulté, une nouvelle approche basée sur l'algorithme *Cognitive-based Adaptive Optimization* (CAO) est proposée et évaluée. L'algorithme CAO est une méthode d'optimisation stochastique adaptative, récemment proposé par Kosmatopoulos dans [61], [63]. Une propriété fondamentale de cette approche réside dans le partage des caractéristiques de convergence des algorithmes

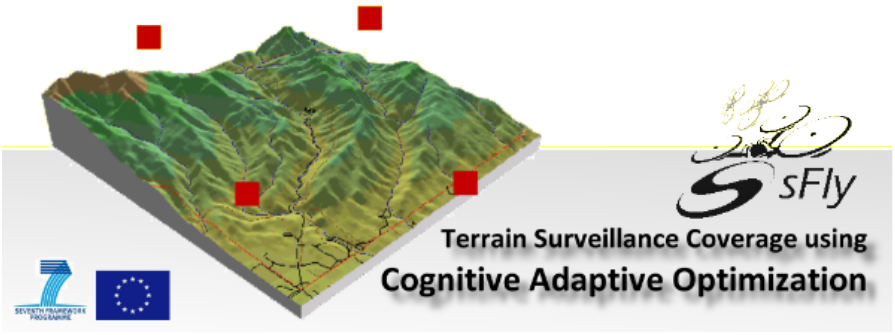


Figure A.3: Le travail développé dans cette thèse s'inscrit dans le cadre du projet européen sFly.

de descente de gradient avec contraintes, qui exigent une connaissance parfaite de la morphologie du terrain. Des arguments mathématiques rigoureux et des simulations étendues prouvent que l'approche proposée fournit une méthodologie évolutive et efficace qui intègre toutes les particulières limites et contraintes physiques et est capable de diriger les robots à un configuration qui (localement) optimise la couverture du terrain.

A.1.3 Organisation et Contributions de la thèse

Le défi de cette thèse était de développer une méthodologie efficace et adaptative pour effectuer une surveillance coopérative d'un terrain très générique et complexe en utilisant une équipe de micro-véhicules aériens. Les difficultés de ce défi ne sont pas seulement de résoudre le complexe problème mathématique, mais elle sont aussi liées à des contraintes fortes que l'application finale réelle exige: une faible complexité de calcul, le manque d'informations sur la surface à couvrir, des éventuelles capacités limitées de communication, et ainsi de suite. Pour aborder ces défis, nous avons adopté un nouvel algorithme d'optimisation stochastique et adaptatif. Toutes les étapes, des les simulations dans des régions 2D jusqu'à la mise en œuvre sur une équipe d'hélicoptères réels ont été effectuées. En outre, pour un critère particulier de couverture, a été mis au point aussi une version distribuée de l'algorithme, ce qui nous permet d'inclure des contraintes de communication et d'éviter tous les problèmes qu'une approche centralisée peut générer.

Le reste de cette thèse est organisé comme suit. Dans le prochain chapitre, nous introduisons le concept d'optimisation stochastique et nous présentons les approches les plus populaires dans la littérature. Ceci permettra au lecteur de mieux comprendre les problématiques principales d'une telle approche. Dans le Chapitre 3, l'algorithme CAO est introduit et ex-

pliqué en détail. En particulier, nous allons fournir une preuve de la convergence vers un minimum local. Les Chapitres 4 et 5 présentent les principaux résultats pour le problème de couverture pour le cas 2D et 3D, respectivement. En suit, dans le Chapitre 6, les résultats expérimentaux obtenus en utilisant des données réelles fournies par des hélicoptères sont présentés et discutés. Le dernier chapitre contient les résultats qui correspondent à un problème différent de navigation dans un environnement avec des personnes. Le but de ce travail final est de montrer la généralité de l'approche proposée. Conclusions et travaux futurs concluent cette thése.

A.2 Optimisation Stochastique

Chaque problème traité dans cette thèse a été abordé comme un problème d'optimisation. L'optimisation pourrait être définie comme la science de déterminer les meilleures solutions à certains problèmes mathématiquement définis, qui sont souvent des modèles de la réalité physique. Dans la pratique, cela signifie maximiser ou minimiser une fonction en choisissant des valeurs d'entrée parmi un ensemble autorisé. Formellement, nous pouvons exprimer ce concept comme suit:

$$x^* \equiv \underset{x \in \mathcal{S}}{\operatorname{argmin}} J(x) = x^* \in \mathcal{S} : J(x^*) \leq J(x) \quad \forall x \in \mathcal{S}\,, \tag{A.1}$$

où x est un vecteur à p-dimensions de paramètres et $\mathcal{S} \subseteq \mathbb{R}^p$. Ce problème implique l'étude des critères d'optimalité pour les problèmes traités, la détermination des méthodes algorithmiques por obtenir une solution, l'étude de la structure de ces méthodes et l'expérimentation informatique de cette méthode à la fois avec des conditions d'essai et avec des problèmes de la vie réelle. Il y a une gamme extrêmement variée d'applications pratiques. On parle d'optimisation stochastique, quand dans la recherche de la solution optimale il y a, d'une certaine façon constructive, un caractère aléatoire. L'optimisation stochastique joue un rôle important dans l'analyse, la conception et l'exploitation de systèmes modernes. Les méthodes pour l'optimisation stochastique fournissent un moyen de faire face à des cas où que de l'information affectée par le bruit est disponible et aussi de faire face à des modèles ou des systèmes qui sont fortement non linéaires, de haute dimension, ou autrement inappropriés pour les méthodes d'optimisation déterministe. Les algorithmes d'optimisation stochastique ont une large application à des problèmes de statistique, d'informatique, d'ingénierie, et de business. Aujourd'hui, les algorithmes employant une certaine forme d'optimisation stochastique sont devenus largement disponibles et exploitées. Les applications spécifiques comprennent par exemple les finances (prise de décisions d'investissement à court et à long terme

afin d'augmenter les profits), l'ingénierie aérospatiale, la médecine, et le trafic (réglage de la synchronisation des signaux dans un réseau de trafic) et, comme dans cette thèse, la robotique mobile.

Le but de ce chapitre est de présenter le problème général qu'un algorithme d'optimisation stochastique est appelé à résoudre et de décrire les algorithmes entre les plus célèbres présents dans la littérature.

En particulier, nous avons fourni une formulation mathématique du problème et de nombreux aspects fondamentaux, et sont discutés les problématiques et les limites de ces types de problèmes d'optimisation. En suite, une brève revue des algorithmes les plus courants est présentée, pour permettre au lecteur de mieux comprendre les possibles stratégies existantes pour trouver une solution. Un autre objectif de ce chapitre est de mettre l'accent sur les caractéristiques qui différencient principalement les algorithmes, en particulier pour les implémentations réelles: les concepts tels que l'évolutivité, l'adaptabilité, le nombre de mesures (ou déplacements des robots) sont d'une importance fondamentale dans la robotique mobile coopérative. Une attention particulière est accordée à l'algorithme SPSA, une approche très proche à l'algorithme que nous proposons, décrivant ses propriétés et ses avantages. Dans le prochain chapitre, nous présenterons en détail l'algorithme d'optimisation stochastique que nous avons adopté dans cette thèse, les motivations de ce choix et ses principales propriétés de convergence.

A.3 L'Algorithme CAO

Dans ce chapitre, nous décrivons en détail l'algorithme que nous utilisons pour obtenir les principaux résultats de cette thèse: l'algorithme d'optimisation cognitive et adaptative (CAO). Nous montrons aussi comment cette approche peut être adaptée et étendue de manière appropriée afin qu'elle soit applicable au problème de couverture traitées dans cette thèse avec une équipe de robots. La méthodologie CAO, qui a été récemment introduite par Kosmatopoulos dans [63], [60], est un algorithme d'optimisation stochastique qui possède la capacité de gérer efficacement des problèmes d'optimisation pour lesquels une expression analytique de la fonction à optimiser n'est pas disponible, mais les valeurs numériques de cette fonction sont disponibles à chaque itération de l'algorithme. En conséquence, il convient parfaitement pour le problème de surveillance optimale dans des environnements non-convexes, où la forme analytique de la fonction à optimiser est inconnue, mais la fonction est disponible pour la mesure (grâce aux capteurs des robots) pour chaque configuration des robots. Dans ce chapitre, nous discutons également les propriétés de convergence de l'algorithme, en four-

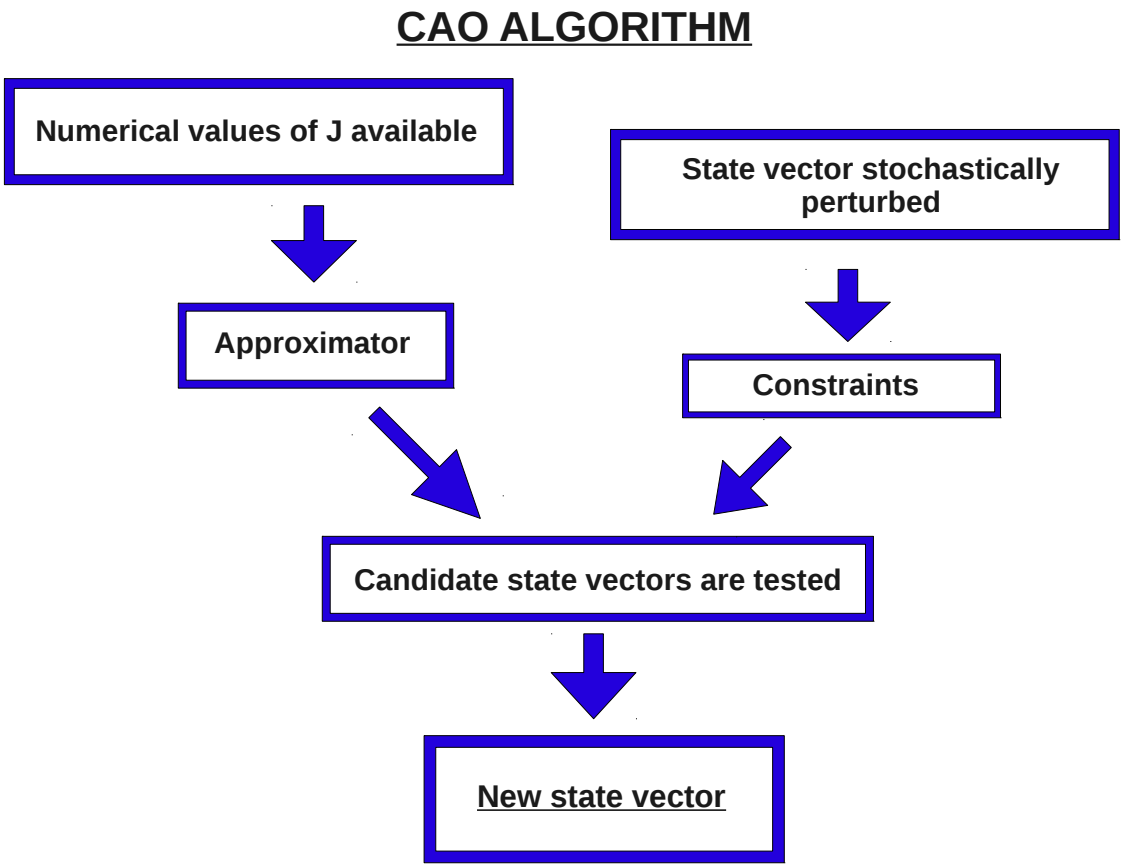


Figure A.4: Principales étapes de l'algorithme CAO.

nissant des démonstrations rigoureuses. Les principales étapes de l'algorithme CAO sont résumées dans la Fig. A.4.

Il doit être souligné que l'algorithme CAO présenté ici est une extension des versions présentées et analysées dans [63, 60]. La principale différence est que, alors que dans ces œuvres les auteurs se penchent sur la version sans contrainte du problème (3.4), dans la présente thèse l'approche CAO [63, 60] doit être étendue afin qu'elle prenne efficacement en compte les contraintes (3.3). Pour ce faire, l'approche en [63, 60] a été modifié avec l'addition d'un mécanisme de projection spéciale, mais pourtant simple. Le théorème 1 établit que l'introduction d'un tel mécanisme de projection ne détruit pas les intéressantes propriétés de la version sans contrainte. Ce résultat est basé sur le fait que l'algorithme CAO utilisé dans cette thèse est avérée être un algorithme de descente de gradient projeté, tandis que les algorithmes en [63, 60] ont été créés pour être approximativement des algorithmes de descente de gradient sans contraintes.

Nous avons finalement mentionné que l'approche CAO étend le populaire Simoultaneous Perturbation Stochastic Approximation (SPSA) algorithme, largement décrit dans le chapitre

précédent. La différence entre le SPSA et l'approche CAO est que le SPSA emploie une approximation du gradient d'une fonction objectif approprié en utilisant uniquement les données les plus récentes disponibles, tandis que l'approche CAO emploie un approximateur linéaire dans les paramètres qui intègre des informations obtenues dans une fenêtre de temps passées, spécifiée par l'utilisateur. En suite il utilise le concept de perturbations random pour optimiser efficacement la fonction inconnue. Les évaluations comparatives qui ont été effectuées sur les problèmes d'optimisation complexes ont montré que le CAO présente des propriétés de convergence meilleurs que SPSA [61, 63, 60]. Par ailleurs, il a été demontré que le CAO a des caractéristiques de convergence (locale) satisfaisantes dans des problèmes particuliers lorsque le SPSA omis de fournir des solutions convergentes pour tout choix de ses paramètres, [61, 60].

Il est important de noter que tant la CAO et la SPSA ne créent pas une approximation ou une estimation des caractéristiques environnementales, comme par exemple l'emplacement des obstacles et sa géométrie, mais plutôt, ils produisent en ligne uniquement une approximation locale de la fonction coût inconnu que les robots sont appelés à optimiser. Pour cette raison, ils ont besoin de schémas d'approximation simples.

Cette nouvelle méthode stochastique d'optimisation est utilisée pour résoudre tous les principaux problèmes considérés dans cette thèse. Pour cette raison, une description précise de toutes les étapes et une preuve mathématique rigoureuse de ses propriétés de convergence ont été fournies. Dans les chapitres suivants, nous allons présenter et analyser les différents critères de couverture possibles, qui sont les fonctions objectives que l'algorithme CAO est appelé à optimiser. En particulier, dans le chapitre suivant, nous considérons le cas le plus facile: nous considérons une région 2D et nous montrons les premiers résultats des simulations obtenus en utilisant la méthode proposée. Nous avons finalement souligné que cet algorithme, comme il est possible de le voir dans la description fournie dans le présent chapitre, peut être appliqué à une gamme très large de problèmes. Le principal problème abordé dans cette thèse, la couverture optimale d'un terrain, est seulement une des applications possibles et dans le chapitre 7 nous allons prouver cette affirmation en abordant un problème complètement différent de robotique avec la même méthode.

A.4 Couverture Optimale - 2D

Le but de ce chapitre est d'appliquer l'algorithme d'optimisation stochastique présenté dans le chapitre précédent pour le principal problème abordé dans cette thèse: la surveillance optimale coopérative. Même si l'objectif ultime consiste à développer une stratégie visant à déployer

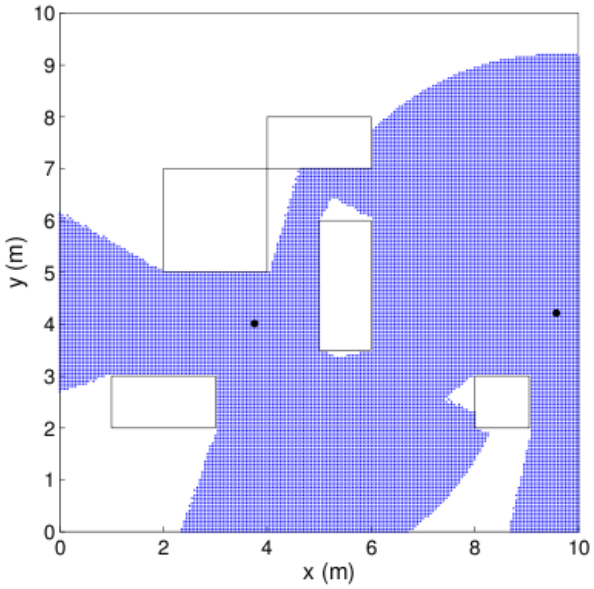


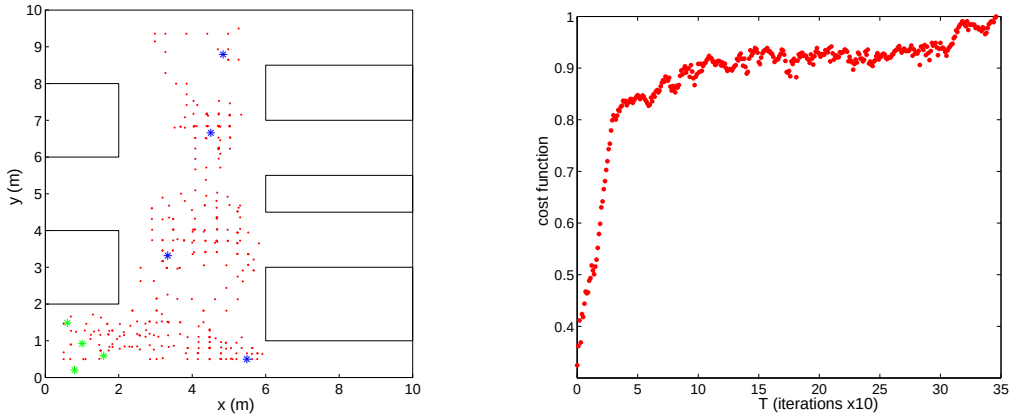
Figure A.5: Exemple de domaine surveillé par deux robots équipés de capteurs visuels omnidirectionnel dans un environnement non-convexe. Les points correspondent à les emplacements des robots, les rectangles représentent les obstacles.

un équipe de micro véhicules aériens dans un environnement réel, nous commençons notre analyse du cas d'une zone 2D. Ce choix est essentiellement dû à deux raisons: la première est que c'est plus facile de commencer notre analyse avec un cas plus simple, bien étudié dans la littérature; d'autre part, il est important de noter qu'une solution pour la région 2D peut être pratique, non seulement pour une équipe de robots terrestres, mais aussi pour une équipe de MAVs, si le terrain est suffisamment plat pour être approximé avec un plan.

Entre les deux critères de couverture énumérés dans l'introduction, ce chapitre se concentre principalement sur le premier: maximiser la partie de la zone que l'équipe est capable de surveiller. Et nous supposons qu'un robot est capable de surveiller tous les points qui sont au sein de son champ de vision (voir Fig. A.5).

Un premier exemple est présenté dans la Fig. A.6(a). Elle montre un scénario typique dans lequel quatre robots doivent couvrir un environnement non convexe. Le comportement de la fonction coût au cours de la tâche est indiqué dans la Fig. A.6(b). Il est possible de voir que le déploiement final est tel que l'équipe est capable de surveiller l'environnement dans son ensemble.

L'algorithme proposé, totalement centralisé dans sa première version, a été en suite distribué. Pour ce faire, nous avons introduit une formulation différente de la fonction objectif qui nous permet de décentraliser l'optimisation. Grâce à cette nouvelle formulation chaque



(a) (b)

Figure A.6: Approche centralisée. Quatre robots avec une distance maximale de surveillance $r = 6m$. Dans la Fig. (a) il est montré le mouvement des robots: les points verts indiquent les positions initiales des robots, les finales sont en bleu, en rouge les trajectoires. En Fig. (b) la fonction objectif $\mathcal{J}(\mathcal{P})$.

robot doit optimiser une fonction différente que l'on peut exprimer comme suit:

$$J_i(\mathcal{P}) = V_i - \sum_{j \neq i} V_i \bigcap V_j \,, \tag{A.2}$$

où V_j est la région de surveillance du j-ième robot (voir Fig. A.7).

Maintenant, au lieu d'une fonction de coût unique, le problème est caractérisé par N fonctions coût, étant N le nombre de robots. L'algorithme CAO est adopté pour maximiser chaque fonction de façon indépendante. Il est important de noter que, à chaque itération, ces nouvelles fonctions d'optimisation ne dépendent que de la position des robots qui ont un champ de vision en intersection avec des autres. Pour cette raison, la méthode distribuée proposée peut également prendre en compte des contraintes de communication.

A.5 Couverture Optimale - 3D

Dans ce chapitre, nous avons étendu les résultats présentés dans le chapitre 4 pour la couverture d'une surface dans un environnement 3D. Tout d'abord, nous avons étendu la solution distribuée fournie dans le chapitre précédent, en montrant qu'elle peut être appliquée aussi dans ce cas. Nous avons également comparé ces résultats avec ceux obtenus en utilisant l'approche centralisée. La comparaison a été complètement satisfaisante puisque les perfor-

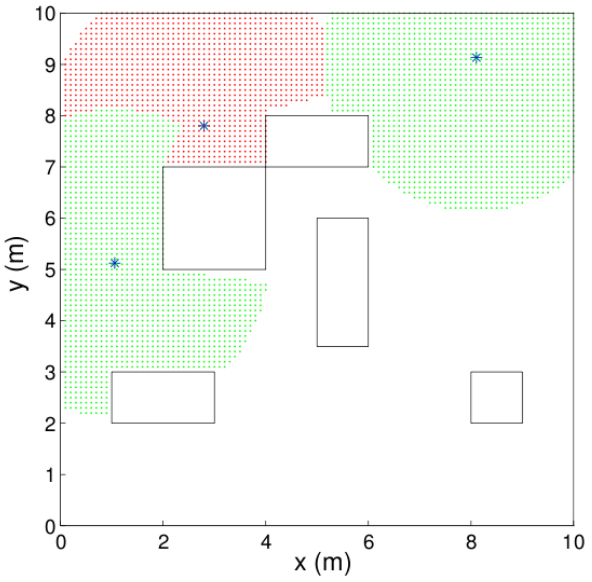


Figure A.7: Algorithme distribué. Exemple de zone surveillée par trois robots équipés de capteur visuel omnidirectionnel dans un environnement non-convexe. Le champ de vision en rouge est l'aire efficace (fonction d'optimisation) surveillée par le robot respectif.

mances des deux méthodes sont très semblables.

Puis, nous avons introduit une nouvelle fonction objective possible, qui essaie de prendre en compte à la fois les critères de couverture précédemment introduits: le problème d'intervention et le problème de visibilité:

$$J(\mathcal{P}) = \int_{q\in\mathcal{V}} \min_{i\in\{1,\dots,M\}} \|x^{(i)} - q\|^2 dq + K \int_{q\in\mathcal{T}-\mathcal{V}} dq \tag{A.3}$$

où K est une constante positive et $\|\cdot\|$ désigne la norme euclidienne. Le premier de ces termes dans l'équation ci-dessus est la fonction de coût pris en compte d'habitude dans de nombreux problèmes de couverture en matière d'environnement 2D connus, liée à la deuxième objectif (réduire au minimum la distance moyenne entre les robots et la sous-zone dont ils sont responsables, voir [20]). Le deuxième terme est lié à la zone invisible sur le terrain ($\int_{q\in\mathcal{T}-\mathcal{V}} dq$ est la partie totale du terrain qui n'est pas visible par l'un des robots).

En général, on ne peut pas optimiser simultanément les deux fonctions, à moins que les fonctions partagent un optimum commun. Par conséquent, l'idée est d'optimiser une fonction objective combinée qui cherche un compromis entre la maximisation de la zone visible et en minimisant la distance des robots vers des points dans l'environnement. L'application de notre algorithme pour ce problème a été présentée et une évaluation du rendement avec plusieurs simulations, réalisées avec complexes environnements simulés, a été remise. Les résultats obtenus montrent que l'algorithme de CAO est en mesure de fournir une solution pour la

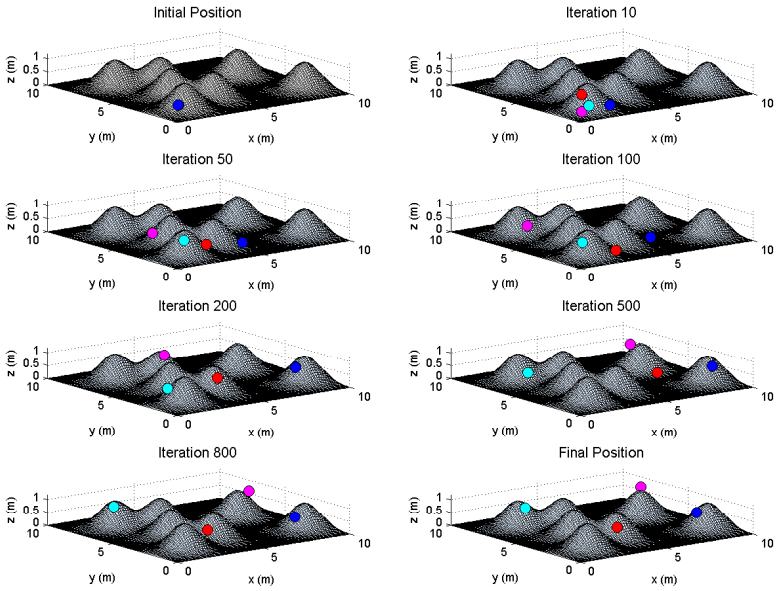


Figure A.8: Instantanés successifs de différentes positions des robots de l'équipe pour $\alpha = 0.3$, dans le cas décrit dans la Section 5.5.1.

couverture de surveillance coopérative pour un environnement 3D totalement arbitraire.

Un exemple de couverture d'un terrain arbitraire en utilisant cette nouvelle formulation du problème est montré dans la Fig. A.8.

Après la validation en utilisant des environnements simulés, dans le chapitre suivant nous présentons les résultats expérimentaux où l'algorithme est testé avec des données réelles fournies par une équipe de MAVs pour une mission de couverture réelle.

A.6 Résultats Expérimentaux

Dans ce chapitre, nous présentons une partie importante de notre travail: l'intégration de notre travail avec les résultats expérimentaux obtenus en utilisant une équipe réel de MAVs. Cette partie du travail est en collaboration également avec les autres partenaires du projet sFly et en particulier avec l'ETH de Zurich (voir [23], [24]).

Initialement, a été présenté et formellement analysé une procédure en deux étapes pour aligner une équipe de véhicules volants pour effectuer une couverture de surveillance. Un algorithme de visual-SLAM suivi les poses d'une caméra tandis que, simultanément, il construit de façon autonome une carte incrémentielle de l'environnement, étant donné que des éléments visuels suffisantes (arbitraires) sont disponibles. Dans les environnements extérieurs

Figure A.9: Trajectoire de vol à travers de la zone de Birmensdorf. La limite de la région d'intérêt est en noir.

Table A.1: Pourcentage de couverture pour des différentes configurations initiales dans le domaine de Birmensdorf.

(% of Coverage)			
Test Case	1	2	3
Initial Configuration	44.49	40.49	56.81
Final Configuration	98.55	99.52	99.56

non préparés, une telle exigence d'avoir des caractéristiques suffisantes est généralement satisfaite. La carte reconstruite de l'environnement est utilisée comme l'entrée pour la deuxième partie de la procédure lorsque la méthodologie CAO est utilisée pour maximiser la zone surveillée par l'équipe de robots aériens. En utilisant cette procédure, plusieurs simulations à la fois dans des régions extérieurs près de Zurich et des espaces intérieurs ont été réalisées.

Un de ces domaines est la région militaire de Birmensdorf à Zurich (see Fig. A.9). Dans le tableau A.6 sont présentés les pourcentages de couverture finale pour des différentes configurations initiales.

Enfin, lors de la démonstration finale du projet, une expérimentation réel avec une équipe de trois micro hélicoptères a été accomplie. L'objectif de la tâche consistait à localiser une victime dans un environnement complexe en plein air, la reproduction d'une recherche typique et la mission de sauvetage. Après une première étape de l'exploration et la cartographie suivant des trajectoires définies manuellement, mais avec un vol basée uniquement sur la vision, la carte a été utilisée comme l'entrée de l'algorithme de CAO pour le calcul des positions finales de couverture optimales. Puis, un hélicoptère survolait sur ces positions et localisait la victime, placée au début de la tâche dans un point quelconque de l'environnement. Cette démonstration avait pour but de combiner le travail de tous les partenaires du projet et de simuler une mission de sauvetage réaliste dans un environnement GPS-nié. La zone

Figure A.10: La zone d'entraînement des pompiers à Zurich: le site choisi pour la démonstration finale du projet sFly.

d'entraînement des pompiers à Zurich a été choisi comme un site idéal pour nos fins (voir Fig. A.10).

Dans la Fig. A.11 il est possible de voir la carte obtenue après l'exploration et le positions finales.

A.7 Navigation entre les Personnes

Comme il a été déjà dit, l'algorithme CAO peut être très utile pour de nombreux problèmes de navigation différents dans des environnements complexes et dynamiques, où un approche adaptative, rapide, à base de capteurs est d'une importance cruciale. Pour le démontrer, dans ce chapitre, nous examinons un problème complètement différent de la couverture de surveillance coopérative et nous montrons comment le même algorithme peut être utilisé pour obtenir une solution dans ce cas aussi. Le problème choisi à cet effet est le suivant: le déplacement d'un robot dans un environnement inconnu et complexe où les personnes se déplacent et interagissent. Le robot doit naviguer en respectant le confort des humains. Nous avons supposé que le robot n'a pas une connaissance a priori sur l'environnement, de sorte que les obstacles

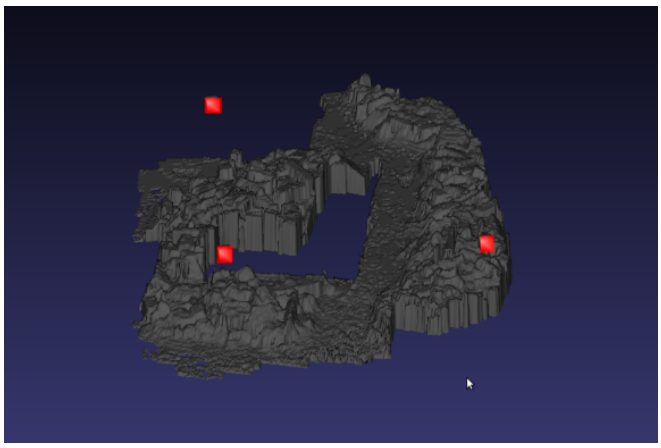


Figure A.11: La carte obtenue par les trois hélicoptères et les positions finales optimales dans une vue 3D.

sont détectés pendant la navigation. Un scénario typique d'un fauteuil roulant se déplaçant entre les humains est représenté en Fig. A.7. Pour obtenir des bons résultats dans de tels environnements, une prédiction sur le mouvement des humains est également cruciale. Pour résoudre tous les problèmes mentionnés ci-dessus, nous introduisons une fonction coût appropriée. Les résultats de ce chapitre ont été présentées dans [97].

Les robots naviguant à proximité de l'homme ou impliqués dans des tâches d'interaction avec les humains, afin d'éviter l'inconfort des personnes, doivent assurer un comportement non seulement sûr, mais aussi compréhensible. Récemment, plusieurs solutions possibles à ce problème ont été proposées [107, 56, 68, 98]. Notre travail est placé dans ce cadre: nous sommes intéressés à guider en toute sécurité un robot dans un environnement inconnu et complexe, où les gens se déplacent et interagissent, et avec l'objectif de respecter le confort des humains. La première étape est de comprendre comment les êtres humains gérent l'espace autour d'eux lors de la navigation et comment leurs décisions affectent le confort des autres. De nombreuses théories psychologiques ont été proposées pour expliquer la relation entre la distance, les comportements visuels et le confort des humains (voir [3] et références incluses). Intuitivement les personnes ne seront plus à l'aise si elles sont approchées à une distance qui est jugée trop étroite: plus grande est l'invasion/intrusion de la zone de l'inconfort et plus la personne ressent de l'excitation. Cette idée simple a été formalisée par l'introduction du concept d'espace personnel, tout d'abord proposé par Hall [42], qui caractérise l'espace autour d'un être humain en termes de confort à l'activité sociale. Dans des conversations informelles, les gens demander un montant de l'espace lié à cette activité. Cet espace est respecté par d'autres personnes et seuls les participants ont permis l'accès à elle, par conséquent, l'intrusion d'un étranger provoque une gêne [53]. On peut supposer que les gens vont s'engager dans un

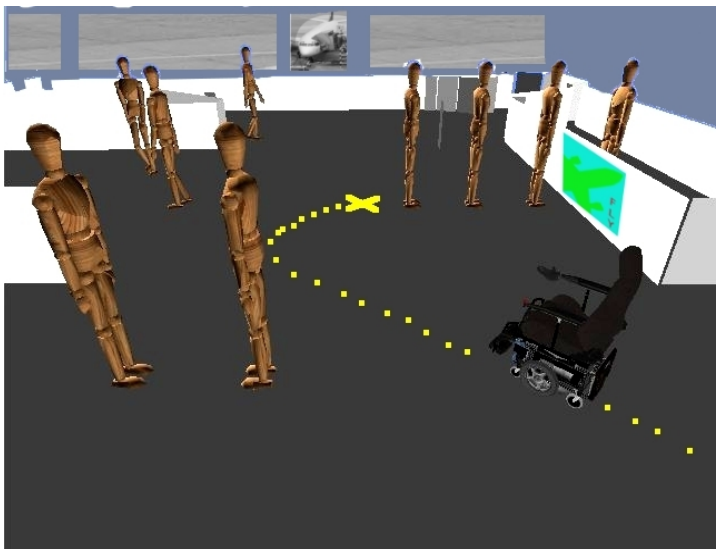


Figure A.12: Un scénario typique de la navigation entre les humains. Un fauteuil roulant est en mouvement dans une zone où les gens discutent et interagissent.

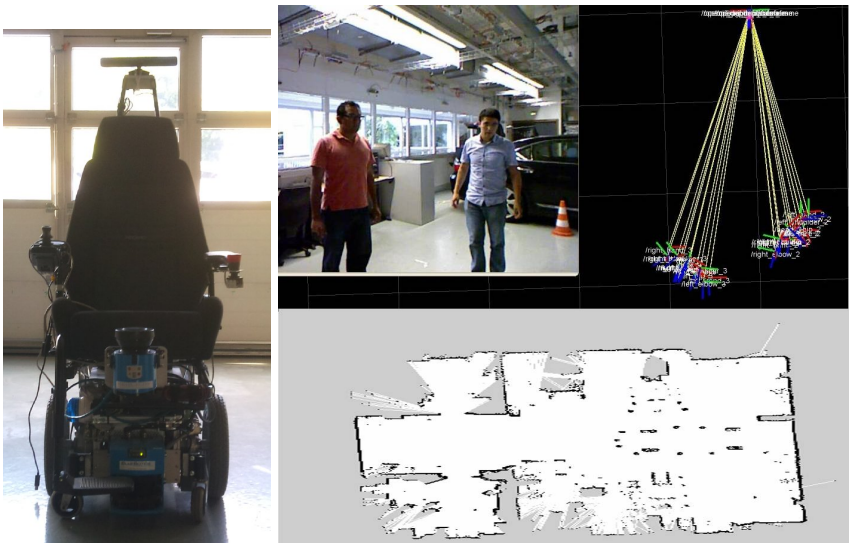
comportement proxémique avec des robots dans une grande partie de la même façon qu'ils interagissent avec d'autres personnes [114].

Nous formulons le problème de la navigation du robot comme un problème d'optimisation où la fonction objectif comprend, en plus de la distance au but, des informations sur le confort des humains. La fonction que nous voulons minimiser est:

$$J = \lambda * (dis(PS) + dis(IPS)) + D(x^{(G)}) \tag{A.4}$$

où λ est un paramètre constant, $dis(PS) + dis(IPS)$ contient l'information sur le confort des personnes et $D(x^{(G)})$ est une fonction de la distance de l'objectif. Dans notre cas, il est tout simplement la distance euclidienne. En utilisant l'algorithme CAO pour générer la trajectoire, nous pouvons aussi obtenir une prédiction indirecte sur le mouvement des personnes, qui est un point très important pour obtenir de bons résultats pour une tâche similaire.

Avec cette application, nous avons voulu montrer comment l'algorithme d'optimisation stochastique exploité dans cette thèse peut être appliqué à des nombreux et très différents problèmes de robotique. En effet, comme nous l'avons déjà dit, grâce à ses nombreux avantages, cette approche est très approprié pour des applications en robotique mobile. Particulièrement en présence d'environnements complexes, dynamiques et incertains. Les résultats obtenus dans ce chapitre sont une forte motivation pour continuer la recherche et pour mettre en œuvre la méthode dans un environnement réel dynamique à l'aide du fauteuil roulant (voir Fig. A.13).



(a) (b)

Figure A.13: Plate-forme expérimentale: en (a) le fauteuil roulant, sur le haut de (b) les données fournies par le kinect, sur le fond la carte finale.

A.8 Conclusions et Travaux à venir

Dans cette thèse, développée dans le cadre du projet européen sFly, nous avons présenté une nouvelle approche basée sur un algorithme d'optimisation adaptatif et stochastique. Le problème de la surveillance coopérative a été étudié avec comme objectif final la surveillance d'une région extérieur en utilisant une équipe de micro-véhicules aériens (MAVs). L'algorithme d'optimisation adopté (CAO), récemment proposée par Kosmatopoulos dans [63, 60], a été convenablement modifié pour être appliqué au problème considéré. En particulier cet algorithme, appliqué à un problème robotique pour la première fois dans cette thèse, a été étendu pour inclure une stratégie pour tenir compte des contraintes, et il a été prouvé que cette nouvelle capacité n'affecte pas les propriétés de convergence. Cette méthode est en mesure de faire face à des problèmes d'optimisation où la fonction objectif est inconnue dans son expression analytique, mais les valeurs numériques sont disponibles pour les mesures, donné une configuration de l'état. Ainsi, il est très utile pour le problème que nous approchions, où le terrain à couvrir peut être inconnu et/ou trop complexe pour être décrit de manière analytique. Plus précisément, l'algorithme proposé présente les avantages clés suivants:

- il ne nécessite aucune connaissance a priori sur l'environnement;
- il fonctionne dans un environnement donné, sans qu'il soit nécessaire de faire une hy-

pothèse concernant sa morphologie;

- il peut intégrer tout type de contraintes;
- sa complexité est faible permettant des implémentations en temps réel;
- il nécessite un faible poids et capteurs à faible coût, ce qui le rend idéal pour des applications en robotique aérienne.

En outre, nous avons proposé une version distribuée de cet algorithme pour un critère de couverture particulière: la maximisation de la surface du terrain que l'équipe est capable de voir. Cette approche est plus proche des applications du monde réel, car il ne souffre pas des inconvénients bien connus d'une méthode centralisée. Per exemple, une approche décentralisée va nous permettre d'inclure des contraintes de communication.

La première étape pour la validation de notre méthode consistait à effectuer des simulations dans des zones 2D, où le problème est plus facile à résoudre et les résultats sont plus compréhensibles. Pour ce premier cas, nous avons également proposé une approche complètement différente, basée sur la méthode du champ de potentiel artificiel et l'algorithme de Lloyd, pour résoudre le problème de couverture de Voronoi d'une région inconnue et non-convexe. Puis, nous avons continué notre enquête en étendant l'algorithme pour le rendre capable de faire face à des environnements 3D. Les premières simulations ont été effectuées en utilisant des environnements complexes simulés et compte tenu de deux différents critères de couverture possibles. Motivés par les bons résultats obtenus dans cette phase, nous avons utilisé des données réelles fournies par un hélicoptère. Ces données ont été recueillies dans des régions extérieures complexes près de Zurich et nous les avons utilisées pour reconstruire une carte de l'environnement, qui a servi comme une entrée pour notre algorithme. Des scénarios incrémentiels sont également testées, où la couverture est réalisée simultanément avec la cartographie. La dernière étape de notre travail a été la mise en œuvre de l'algorithme sur une équipe réel de MAV pour effectuer une mission de couverture. Cette mission de couverture a été réalisée dans une zone extérieure à Zurich dans le but de localiser une victime placée dans un point quelconque de l'environnement.

Enfin, nous avons abordé aussi un autre problème de navigation. Dans ce cas, l'objectif a été la navigation en toute sécurité d'un robot dans un environnement inconnu et complexe où les personnes se déplacent et interagissent. L'objectif du robot a été de respecter le confort des personnes. Aussi pour cette application, les résultats étaient tout à fait satisfaisants et une mise en œuvre sur une plate-forme expérimentale est attendue.

Nous nous attendons à ce que de nombreuses tâches importantes de la robotique mobile pourront être approchées par les algorithmes basés sur l'algorithme CAO. Par exemple:

l'exploration coordonnée, le suivi optimal de cible, etc. Cela est essentiellement dû au fait que l'approche CAO ne nécessite pas une connaissance a priori de l'environnement et elle exige de faibles ressources de calcul. Ces deux questions sont fondamentales en robotique mobile.

Bibliography

[1] P. Agarwal and M. Sharir. Efficient algorithms for geometric optimization. *ACM Computing Surveys*, 30(4):412–458, 1998.

[2] A. Ahmadzadeh, A. Jadbabaie, V. Kumar, and G. Pappas. Multi-uav cooperative surveillance with spatio-temporal specifications. In *45th IEEE Conference on Decision and Control (CDC)*, pages 5293–5298, San Diego, CA, USA, 2006.

[3] J. Aiello. Human spatial behavior. In *Handbook of environmental psychology*. John Wiley & Sons, 1987.

[4] M. Argyle and J. Dean. Eye-contact, distance and affiliation. *Sociometry*, 28(3):289–304, 1965.

[5] M. Baseggio, A. Cenedese, P. Merlo, M. Pozzi, , and L. Schenato. Distributed perimeter patrolling and tracking for camera networks. In *49th IEEE Conference on Decision and Control (CDC)*, Atlanta, GA, USA, 2010.

[6] M. Batalin and G. Sukhatme. Spreading out: A local approach to multi-robot coverage. In *6th International Conference on Distributed Autonomous Robotic System (DARS)*, Fukoka, Japan, 2002.

[7] M. Bazaraa, H. Sherali, and C. Shetty. *Nonlinear programming: theory and algorithms*. John Wiley and Sons, 2006.

[8] D. P. Bertsekas and J. N. Tsitsiklis. Gradient convergence in gradient methods with errors. *SIAM Journal in Optimization*, 10(3):627–642, 2000.

[9] S. Bhatnagar, M. Fu, S. Marcus, and I. Wang. Two-timescale simultaneous perturbation stochastic approximation using deterministic perturbation sequences. *ACM Transactions on Modeling and Computer Simulation*, 13:180–209, 2003.

[10] S. Bhattacharya, N. Michael, and V. Kumar. Distributed coverage and exploration in unknown non-convex environments. In *10th International Symposium on Distributed Autonomous Robots (DARS)*, Lausanne, Switzerland, 2010.

[11] M. Bloesch, S. Weiss, D. Scaramuzza, and R. Siegwart. Vision based mav navigation in unknown and unstructured environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, AK, USA, 2010.

[12] M. Bloesch, S. Weiss, D. Scaramuzza, and R. Siegwart. Vision based mav navigation in unknown and unstructured environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, AK, USA, 2010.

[13] J. Blum. Multidimensional stochastic approximation methods. *The Annals of Mathematical Statistics*, 25(4):737–744, 1954.

[14] V. Borkar and S. Meyn. O.d.e. method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal in Optimization*, 38:447–469, 2000.

[15] A. Breitenmoser, J. Metzger, R. Siegwart, and D. Rus. Distributed coverage control on surfaces in 3d space. In *Proceedings of the IEEE International Conference on Robotics and Intelligent System (IROS)*, Taipei, Taiwan, 2010.

[16] A. Breitenmoser, M. Schwager, J. Metzger, R. Siegwart, and D. Rus. Voronoi coverage of non-convex environments with a group of networked robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Anchorage, AK, USA, 2010.

[17] C. Caicedo-Nunez and M. Zefran. Performing coverage on nonconvex domains. In *IEEE International Conference on Control Applications (CCA)*, pages 1019–1024, San Antonio, TX, USA, 2008.

[18] H. Chen and Z. Xu. 3d map building based on stereo vision. In *IEEE International Conference on Networking, Sensing and Control, ICNSC*, 2006.

[19] H. Choset and P. Pignon. Coverage path planning: The boustrophedon cellular decomposition. In *International Conference on Field and Service Robotics*, 1997.

[20] J. Cortés, S. Martínez, T. Karataş, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004.

[21] K. Dautenhahn, M. Walters, S. Woods, K. L. Koay, C. L. Nehaniv, A. Sisbot, R. Alami, and T. Siméon. How may i serve you?: a robot companion approaching a seated person in a helping context. In *HRI 06: Proceeding of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 172–179. ACM Press, 2006.

[22] J. Dippon and J. Renz. Weighted means in stochastic approximation of of minima. *SIAM Journal of Control and Optimization*, 35:1811–1827, 1997.

[23] L. Doitsidis, A. Renzaglia, S. Weiss, E. Kosmatopoulos, D. Scaramuzza, and R. Siegwart. 3d surveillance coverage using maps extracted by a monocular slam algorithm. In *IEEE International Conference on Robotics and Intelligent System (IROS)*, pages 1661–1667, San Francisco, CA, USA, 2011.

[24] L. Doitsidis, S. Weiss, A. Renzaglia, M. Achtelik, E. Kosmatopoulos, R. Siegwart, and D. Scaramuzza. Optimal surveillance coverage for teams of micro aerial vehicles in gps-denied environments using onboard vision. *Autonomous Robots Special Issue on Micro-UAV Perception and Control*, to appear, 2012.

[25] J. Durham, R. Carli, P. Frasca, and F. Bullo. Dynamic partitioning and coverage control with asynchronous one-to-base-station communication. In *50th IEEE Conference on Decision and Control (CDC)*, pages –, Orlando, FL, USA, 2011.

[26] H. Fang, G. Gong, and M. Qian. Annealing of iterative stochastic schemes. *SIAM journal on control and optimization*, 35:1886, 1997.

[27] D. Fouskakis and D. Draper. Stochastic optimization. *International Statistical Review*, 70:315–349, 2002.

[28] R. Freeman and P. Kokotovic. Inverse optimality in robust stabilization. *SIAM Journal on Control and Optimization*, 34(4):1365–1391, 1996.

[29] M. Fu. Optimization for simulation: theory vs. practice. *INFORMS Journal on Computing*, 14:192–227, 2002.

[30] Y. Gabriely and E. Rimon. Spanning-tree based coverage of continuous areas by a mobile robot. *Ann. Math. Artif. Intell.*, 31:77–98, 2001.

[31] D. Gage. Command control for many-robot systems. In *AUVS-92, the Nineteenth Annual AUVS Technical Symposium*, Hunstville, Alabama, USA, 1992.

[32] A. Ganguli, J. Cortés, and F. Bullo. Maximizing visibility in nonconvex polygons: nonsmooth analysis and gradient algorithm design. In *Proceedings of the American Control Conference (ACC)*, pages 792–797, 2005.

[33] A. Ganguli, J. Cortés, and F. Bullo. Distributed deployment of asynchronous guards in art galleries. In *American Control Conference, 2006*, pages 6–pp, 2006.

[34] A. Ganguli, J. Cortés, and F. Bullo. Visibility-based multi-agent deployment in orthogonal environments. In *Proceedings of the American Control Conference (ACC)*, pages 3426–3431, USA, 2007.

[35] S. Gelfand and S. Mitter. Recursive stochastic algorithms for global optimization in ird. In *29th IEEE Conference on Decision and Control (CDC)*, pages 220–221, 1990.

[36] A. T. GmbH. website. http://www.asctec.de.

[37] A. Gosavi. *Simulation-based Optimization: Parametric Optimization Techniques and Reinforcement Learning*. Kluwer Academic Publishers, Boston, MA, 2003.

[38] D. Gurdan, J. Stumpf, M. Achtelik, K. Doth, G. Hirzinger, and D. Rus. Energy-efficient autonomous four-rotor flying robot controlled at 1 khz. In *IEEE International Conference on Robotics and Automation*, pages 361–366, Roma, Italy, 2007.

[39] A. Gusrialdi, S. Hirche, T. Hatanaka, and M. Fujita. Voronoi based coverage control with anisotropic sensors. In *American Control Conference, 2008*, pages 736–741, 2008.

[40] J. M. H. Strasdat and A. Davison. Real-time monocular slam: Why filter?'. In *it IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 2011.

[41] H. Haddad, M. Khatib, S. Lacroix, and R. Chatila. Reactive navigation in outdoor environments using potential fields. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1232–1237, 1998.

[42] E. Hall. *The hidden Dimension: Man's Use of Space in Public and Private*. The Bodley Head Ltd, London, UK, 1966.

[43] B. Hexsel, N. Chakraborty, and K. Sycara. Coverage control for mobile anisotropic sensor networks. In *IEEE International Conference on Robotics and Automation*, pages 2878–2885, Shanghai, China, 2010.

[44] Y. Ho. On the numerical solutions of stochastic optimization problems. *IEEE Transaction on Automatic Control*, 42:727–729, 1997.

[45] A. Howard, M. Matarić, and G. Sukhatme. An incremental deployment algorithm for mobile robot teams. In *Proceedings of the IEEE International Conference on Robotics and Intelligent System (IROS)*, pages 2849–2854, Lausanne, Switzerland, 2002.

[46] A. Howard, M. Matarić, and G. Sukhatme. Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem. In *6th International Conference on Distributed Autonomous Robotic System (DARS)*, pages 299–308, Fukoka, Japan, 2002.

[47] G. Huang and L. Chen. Convex incremental extreme learning machine. *Neurocomputing*, 70:3056–3062, 2007.

[48] G. Huang, L. Chen, and C. Sew. Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Transaction on Neural Networks*, 17:879–892, 2006.

[49] I. Hussein and D. Stipanovic. Effective coverage control for mobile sensor networks with guaranteed collision avoidance. *IEEE Transactions on Control Systems Technology*, 15(4):642–657, 2007.

[50] P. Ioannou and J. Sun. Stable and robust adaptive control. *Englewood Cliffs, NJ: Printice Hall,*, 2, 1995.

[51] L. Jin, M. Gupta, and P. N. Nikiforuk. Neural networks and fuzzy basis functions for functional approximation. *Fuzzy Logic and Intelligent Systems*, 3:17–67, 1995.

[52] I. Kamon, E. Rimon, and E. Rivlin. Tangentbug: A range-sensor-based navigation algorithm. *The International Journal of Robotics Research*, 17(9):934–953, 1998.

[53] A. Kendon. Spacing and orientation in co-present interaction. In *Development of Multimodal Interfaces: Active Listening and Synchrony*, volume 5967 of *Lecture Notes in Computer Science*, pages 1–15. Springer Berlin / Heidelberg, 2010.

[54] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. *International Journal of Robotics Research*, 5(1):90–98, 1986.

[55] J. Kiefer and J. Wolfowitz. Stochastic estimation of the maximum of a regression function. *The Annals of Mathematical Statistics*, 23(3):462–466, 1952.

[56] R. Kirby, R. Simmons, and J. Forlizzi. Companion: A constraint-optimizing method for person acceptable navigation. *The 18th IEEE International Symposium on Robot and Human Interactive Communication*, 2009.

[57] K. Kitazawa and T. Fujiyama. Pedestrian vision and collision avoidance behavior: Investigation of the information process space of pedestrians using an eye tracker. In *Pedestrian and Evacuation Dynamics 2008*, chapter 7, pages 95–108. Springer, Berlin, Heidelberg, 2010.

[58] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *Proceedings of the Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR07)*, 2007.

[59] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *International Symposium on Mixed and Augmented Reality*, pages 225–234, 2007.

[60] E. Kosmatopoulos and A. Kouvelas. Large-scale nonlinear control system fine-tuning through learning. *IEEE Transactions Neural Networks*, 20(6):1009–1023, 2009.

[61] E. Kosmatopoulos, M. Papageorgiou, A. Vakouli, and A. 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.

[62] E. Kosmatopoulos, M. Polycarpou, M. Christodoulou, and P. Ioannou. High-order neural network structures for identification of dynamical systems. *IEEE Transactions Neural Networks*, 6:422–431, 1995.

[63] E. B. Kosmatopoulos. An adaptive optimization scheme with satisfactory transient performance. *Automatica*, 45(3):716–723, 2009.

[64] H. Kushner. Asymptotic global behavior for stochastic approximation and diffusions with slowly decreasing noise effects: Global minimization via monte carlo. *SIAM Journal on Applied Mathematics*, pages 169–185, 1987.

[65] H. Kushner and D. Clark. *Stochastic Approximation Methods for Constrained and Unconstrained Systems*, volume 6. Springer-Verlag Berlin, New York, 1978.

[66] H. Kushner and G. Yin. *Stochastic Approximation Algorithms and Applications*. Springer-Verlag, New York, 1997.

[67] I. Kweon and T. Kanade. High resolution terrain map from multiple sensor data. In *Intelligent Robots and Systems' 90.'Towards a New Frontier of Applications', Proceedings. IROS'90. IEEE International Workshop on*, pages 127–134, 1990.

[68] C. P. Lam, C. T. Chou, K. H. Chiang, and L. C. Fu. Human-centered robot navigation, towards a harmoniously human-robot coexisting environment. *IEEE Transactions on Robotics*, 27(1), 2011.

[69] J. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Boston, MA, 1991.

[70] K. Laventall and J. Cortés. Coverage control by robotic networks with limited-range anisotropic sensory. In *American Control Conference, 2008*, pages 2666–2671, 2008.

[71] R. Lewis, V. Torczon, and M. Trosset. Direct search methods: Then and now. *Journal Comput. Appl. Math.*, 124:191–207, 2000.

[72] M. Likhachev and R. Arkin. Robotic comfort zones. In *Proceedings of SPIE: Sensor Fusion and Decentralized Control in Robotic Systems III Conference*, volume 4196, pages 27–41, 2000.

[73] S. Lloyd. Least-squares quantization in pcm. *IEEE Trans. Inform. Theory*, IT-28:129–137, Jan. 1982.

[74] J. Luna, R. Fierro, C. Abdallah, and J. Wood. An adaptive coverage control algorithm for deployment of nonholonomic mobile sensors. In *49th IEEE Conference on Decision and Control (CDC)*, pages 1250–1256, Atlanta, GA, USA, 2010.

[75] V. Maiorov and R. Meir. Approximation bounds for smooth functions in $c(r^d)$ by neural and mixture networks. *IEEE Transactions on Neural Networks*, 9(5):969–978, 1998.

[76] J. Maryak and D. Chin. Efficient global optimization using spsa. In *Proceedings of the American Control Conference (ACC)*, volume 2, pages 890–894, 1999.

[77] J. Maryak and D. Chin. Global random optimization by simultaneous perturbation stochastic approximation. *IEEE Transactions on Automatic Control*, 53(3):780–783, 2008.

[78] J. Matyas. Random optimization. *Automation and Remote Control*, 26:244–251, 1965.

[79] N. Michael, E. Stump, and K. Mohta. Persistent surveillance with a team of mavs. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2708–2714, San Francisco, CA, USA, 2011.

[80] J. L. Ny and G. Pappas. Sensor-based robot deployment algorithms. In *49th IEEE Conference on Decision and Control (CDC)*, pages 5486–5492, Atlanta, GA, USA, 2010.

[81] J. L. Ny, A. Ribeiro, and G. Pappas. Robot deployment with end-to-end wireless communication constraints. In *50th IEEE Conference on Decision and Control (CDC)*, pages –, Orlando, FL, USA, 2011.

[82] A. Okabe, B. Boots, K. Sugihara, and S. Chiu. *Spatial Tessellation: Concepts and Application of Voronoi Diagrams*. Wiley Series in Probability and Statistics. Wiley, New York, USA, 2000.

[83] D. Olguin, P. Gloor, and A. Pentland. Capturing individual and group behavior with wearable sensors. In *AAAI Spring Symposium on Human Behavior Modeling*, 2009.

[84] J. O'Rourke. *Art gallery theorems and algorithms*, volume 57. Oxford University Press Oxford, 1987.

[85] L. Pimenta, V. Kumar, R. Mesquita, and G. Pereira. Sensing and coverage for a network of heterogeneous robots. In *47th IEEE Conference on Decision and Control*, pages 3947–3952, Cancun, Mexico, 2008.

[86] S. Poduri and G. Sukhatme. Constrained coverage for mobile sensor networks. In *IEEE International Conference on Robotics and Automation*, pages 165–172, New Orleans, LA, USA, 2004.

[87] M. Polycarpou and P. Ioannou. Identification and control of nonlinear systems using neural network models: Design and stability analysis. Technical Report 91 -09-01, University of Southern California, Dept. Electrical Engineering - Systems, 1991.

[88] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. ROS: an open-source Robot Operating System. In *ICRA Workshop on Open Source Software*, Kobe, Japan, 2009.

[89] D. Raimondo, N. Kariotoglou, S. Summers, and J. Lygeros. Probabilistic certification of pan-tilt-zoom camera surveillance systems. In *50th IEEE Conference on Decision and Control (CDC)*, Orlando, FL, USA, 2011.

[90] A. Renzaglia, L. Doitsidis, A. Martinelli, and E. Kosmatopoulos. Adaptive-based, scalable design for autonomous multi-robot surveillance. In *Proceedings of the 49th IEEE Conference on Decision and Control (CDC)*, pages 4618–4624, Atlanta, GA, USA, 2010.

[91] A. Renzaglia, L. Doitsidis, A. Martinelli, and E. Kosmatopoulos. Cognitive-based adaptive control for cooperative multi-robot coverage. In *Proceedings of the IEEE*

International Conference on Robotics and Intelligent System (IROS), pages 3314–3320, Taipei, Taiwan, 2010.

[92] A. Renzaglia, L. Doitsidis, A. Martinelli, and E. Kosmatopoulos. Adaptive-based distributed cooperative multi-robot coverage. In *Proceedings of the American Control Conference (ACC)*, pages 468–473, San Francisco, CA, USA, 2011.

[93] A. Renzaglia, L. Doitsidis, A. Martinelli, and E. Kosmatopoulos. Multi-robot 3d coverage of unknown terrains. In *50th IEEE Conference on Decision and Control (CDC)*, pages 2046–2051, Orlando, FL, USA, 2011.

[94] A. Renzaglia, L. Doitsidis, A. Martinelli, and E. Kosmatopoulos. Multi-robot 3d coverage of unknown areas. *The International Journal of Robotics Research*, 2012.

[95] A. Renzaglia and A. Martinelli. Distributed coverage control for a multi-robot team in a non-convex environment. In *IEEE IROS09 3rd Workshop on Planning, Perception and Navigation for Intelligent Vehicles*, pages 76–81, St Louis, MO, USA, 2009.

[96] A. Renzaglia and A. Martinelli. Potential field based approach for coordinate exploration with a multi-robot team. In *8th IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR)*, Bremen, Germany, 2010.

[97] J. Rios-Martinez, A. Renzaglia, A. Spalanzani, A. Martinelli, and C. Laugier. Navigating between people: a stochastic optimization approach. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages –, St. Paul, MN, USA, 2012.

[98] J. Rios-Martinez, A. Spalanzani, and C. Laugier. Understanding human interaction for probabilistic autonomous navigation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[99] H. Robbins and S. Monro. A stochastic approximation method. *Annals of Mathematical Statistics*, 22:400–407, 1951.

[100] H. Robbins and D. Siegmund. A convergence theorem for nonnegative almost supermartingales and some applications. *Optimizing methods in statistics*, pages 233–257, 1971.

[101] A. M. S. Lacroix, I. Jung. Digital elevation map building from low altitude stereo imagery. In *9th International Symposium on Intelligent Robotic Systems*, 2001.

[102] P. Sadegh. Constrained optimization via stochastic approximation with a simultaneous perturbation gradient approximation. *Automatica*, 33:889–892, 1997.

[103] M. Schwager, F. Bullo, D. Skelly, and D. Rus. A ladybug exploration strategy for distributed adaptive coverage control. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2346–2353, Pasadena, CA, USA, 2008.

[104] M. Schwager, B. Julian, and D. Rus. Optimal coverage for multiple hovering robots with downward facing camera. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3515–3522, Kobe, Japan, 2009.

[105] M. Schwager, J. McLurkin, and D. Rus. Distributed coverage control with sensory feedback for networked robots. In *Proceedings of Robotics: Science and Systems*, pages 49–56, Philadelphia, PA, USA, 2006.

[106] T. Shermer. Recent results in art galleries. In *IEEE Proceedings*, volume 80, pages 1384–1399, 1992.

[107] E. Sisbot, L. Marin-Urias, R. Alami, and T. Simeon. A human aware mobile robot motion planner. *IEEE Transactions on Robotics*, 23, 2007.

[108] J. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992.

[109] J. Spall. Accelerated second-order stochastic optimization using only function measurements. In *36th IEEE Conference on Decision and Control (CDC)*, volume 2, pages 1417–1424, 1997.

[110] J. Spall. An overview of the simultaneous perturbation method for efficient optimization. *Johns Hopkins APL Technical Digest*, 19(4):482–492, 1998.

[111] J. Spall. *Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control*. Wiley, Hoboken, NJ, 2003.

[112] J. Spall. Stochastic optimzation. In *Handbook of Computational Statistics*, pages 169–197. Springer-Verlag, New York, 2004.

[113] M. Styblinski and T. Tang. Experiments in nonconvex optimization: stochastic approximation with function smoothing and simulated annealing. *Neural Networks*, 3(4):467–483, 1990.

[114] L. Takayama and C. Pantofaru. Influences on proxemic behaviors in human-robot interaction. *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

[115] S. Thrun, W. Burgard, and D. Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, volume 1, pages 321–328, 2000.

[116] R. Triebel, P. Pfaff, and W. Burgard. Multi-level surface maps for outdoor terrain mapping and loop closing. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2276–2282, 2006.

[117] J. Urrutia. Art gallery and illumination problems. *Handbook of computational geometry*, pages 973–1027, 2000.

[118] M. Valera and S. Velastin. Intelligent distributed surveillance systems: a review. *IEEE Proceedings - Vision, Image, and Signal Processing*, 152(2):192–204, 2005.

[119] S. Weiss, M. Achtelik, L. Kneip, D. Scaramuzza, and R. Siegwart. Intuitive 3d maps for mav terrain exploration and obstacle avoidance. *Journal of Intelligent & Robotic Systems*, 61:473–493, 2011.

[120] S. Weiss, D. Scaramuzza, and R. Siegwart. Monocular-slam-based navigation for autonomous micro helicopters in gps-denied environments. *Journal of Field Robotics*, 28(6):854–874, 2011.

[121] S. Weiss and R. Siegwart. Real-time metric state estimation for modular vision-inertial systems. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4531–4537, Shanghai, China, 2011.

[122] G. Yin. Rates of convergence for a class of global stochastic optimization algorithms. *SIAM Journal on Optimization*, 10:99–120, 1999.