Coordination of Cooperative Multi-Robot Teams

Vom Fachbereich
Elektrotechnik und Informationstechnik
der Technischen Universität Darmstadt
zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.)
genehmigte Dissertation

von

Lukas Klodt, M.Sc.

geboren am 20. September 1986
in Rotenburg a. d. Fulda, Hessen

Referent:	Prof. Dr.-Ing. J. Adamy
Korreferent:	Assoc. Prof. J. Faigl, Ph.D.
Tag der Einreichung:	19. September 2017
Tag der mündlichen Prüfung:	30. November 2017

D17
Darmstadt 2017

Erklärung laut §9 PromO

Ich versichere hiermit, dass ich die vorliegende Dissertation allein und nur unter Verwendung der angegebenen Literatur verfasst habe. Die Arbeit hat bisher noch nicht zu Prüfungszwecken gedient.

Darmstadt, den 19. September 2017 Lukas Klodt

Acknowledgments

First of all, I would like to thank my supervisors Prof. Jürgen Adamy and Volker Willert for their constant support and high degree of freedom to work in a self-organized manner. To Jürgen Adamy I am grateful especially for leading the Control Methods and Robotics Lab, taking care of financial safety by providing for travel expenses and a follow-up scholarship, his trust and patience during the last phase, where he offered ideas for future undertakings. Volker Willert was always open to discuss ideas regarding the contents of my thesis, to listen in times of difficulty or crisis, when my motivation was low, and generally held up a positive attitude in the whole group.

As this work was mainly funded by the Deutsche Forschungsgemeinschaft within a Research Training Group (Graduiertenkolleg GRK 1362), I am grateful for the financial support and the other opportunities arising from it. Scientific exchange was facilitated, not only with colleagues from other departments but also universities from all over the world through talks with invited speakers and a Summer School in 2014, where I was part of the organizing committee. The GRK also led to contact with my co-referee Assoc. Prof. Jan Faigl from the Czech Technical University in Prague, whom I invited to Darmstadt for a talk about multi-goal path planning. I want to thank him for the fruitful discussions we had during his visit and when meeting at conferences.

I am grateful for the colleagues at our lab, providing different perspectives and resources of expertise in various fields as well as company during free-time activities. Dominik Haumann introduced me to Linux and his research, giving me something to start with and providing excellent collaboration for the first publication, which would have taken much longer without his support. I also thank my former students who gave me the opportunity to practice and learn supervising others in their scientific work.

Special thanks go to Birgit Heid for always being the ray of sunlight and warmth no matter the weather and providing all the office assistance one could ask for, Susanne Muntermann for her efforts, endurance and interest in my office break time activities, Sylvia Gelman for the technical support and collaboration as system administrator, Valentina Ansel for the fancy robot picture, shared activities like swimming in beautiful lakes and visiting Jochen after he left the lab.

The highest gratitude goes to my parents. Without their unconditional support during my whole time at the university, none of this would have been possible. Thanks to my sister, who gifted me with many heartwarming moments especially in the last year and my brother showing me new horizons I would not have discovered otherwise.

Darmstadt, September 2017 Lukas Klodt

Contents

List of Figures x

Abstract xii

1 Introduction **1**
1.1 Motivation 2
1.2 Cooperative Robotics and Multi-Agent Systems 3
1.3 Goals and Contributions 10
1.4 Thesis Outline 12

2 Mathematical Foundations **13**
2.1 Graph Theory 13
2.2 Computational Geometry 14
2.2.1 Sets and Environments 14
2.2.2 Distance Metrics 16
2.2.3 Visibility 16
2.3 Dynamical Systems 16
2.3.1 Nonlinear Systems 16
2.3.2 Linear Time Invariant Systems 19

3 UAV-UGV Cooperation **20**
3.1 Introduction and Literature Review 20
3.2 Problem Formulation 22
3.3 Dynamic Coverage 24
3.3.1 Original Approach 25
3.3.2 Extensions 26
3.3.3 Simulation Example 29
3.4 Tracking Control 30
3.5 Complete Motion Control 33
3.5.1 Combined Coverage and Tracking 33
3.5.2 Virtual Point Tracking 35
3.6 Simulation Results 37
3.7 Discussion 38

4 Coordination in Multi-Robot Exploration 42
4.1 Introduction 42
  4.1.1 Literature Review 44
  4.1.2 Contribution 46
4.2 Problem Statement 47
  4.2.1 Setting and Assumptions 47
  4.2.2 Cost Function 48
  4.2.3 Approximating the Cost Function 49
4.3 Pairwise Optimization 50
  4.3.1 Searching the Solution Space 51
  4.3.2 Analysis of the Assignment Procedure 53
4.4 Exploration Framework 55
  4.4.1 Frontier Segmentation 55
  4.4.2 Robot Motion and Target Selection 55
  4.4.3 Distributed Implementation 56
4.5 Results 57
  4.5.1 Evaluation of Communication Requirements 57
  4.5.2 Comparison with Related Work 58
4.6 Discussion 61

5 Coverage Control in Nonconvex Environments 63
5.1 Introduction 63
  5.1.1 Literature Review 64
  5.1.2 Contribution 67
5.2 Problem Formulation 67
  5.2.1 Voronoi Partitions 67
  5.2.2 Lloyd Algorithm 68
  5.2.3 Coverage in Convex Environments 70
5.3 Extension to Nonconvex Environments 73
  5.3.1 Visibility 73
  5.3.2 Delta-contraction 74
  5.3.3 Derivation of the Objective Function 74
  5.3.4 Trajectories 79
  5.3.5 Limited Sensing Range 80
5.4 Results 81
  5.4.1 Simulations 81
  5.4.2 Comparison 85
5.5 Discussion 86

6 Summary and Outlook 87
6.1 Overview 87
6.2 UAV-UGV Cooperation 88
6.3 Multi-Robot Exploration 89
6.4 Coverage 90
6.5 Complete System Architecture 91

Bibliography 93

List of Figures

1.1 Different examples for multi-agent control - part 1 6
1.2 Feedback loop for active sensing tasks 8
1.3 Different examples for multi-agent control - part 2 9
1.4 Stages of a search and rescue mission as outlined in this thesis 10

2.1 A graph and two spanning trees 14
2.2 Environments and visibility 15
2.3 Attractivity and Lyapunov stability, resembling [2] 17

3.1 Example of an air and ground vehicle constellation 21
3.2 Values of the UAV sensory function $S_a(\boldsymbol{p}_a, \boldsymbol{q})$ for different distances of an observed point $\boldsymbol{q}$ 23
3.3 Weighted vectors pointing to uncovered areas illustrating the mechanics of (3.6) 26
3.4 The density function as in (3.9) 27
3.5 Trajectories and covered area for coverage behavior without tracking 29
3.6 Velocity and coverage error for different control gains corresponding to the trajectories in Figure 3.5 30
3.7 Block diagram with the converted dynamic controller for the UAV 32
3.8 Different weighting functions for the combined coverage and tracking control law 35
3.9 UAV-UGV distance for the different weighting functions shown in Figure 3.8 36
3.10 Vector components illustrating the virtual point tracking . . 36
3.11 Simulation results for scenario 1 39
3.12 Simulation results for scenario 2 40

4.1 Frontier-based exploration with a single robot 43
4.2 Comparison of greedy and TSP-based goal selection 44
4.3 Minimum spanning trees between the robot positions and all assigned target points 50

4.4 Frontier segments in different colors, segmented according to (4.7) 56
4.5 Scenarios for evaluation 57
4.6 Exploration times with increasing amount of communication 59
4.7 Exploration times compared for scenario A 60
4.8 Exploration times compared for scenario B 61
4.9 Exploration times compared for scenario C 61

5.1 Simple example of a coverage task 64
5.2 Two cases where the original algorithm fails in nonconvex environments 65
5.3 Example of a Voronoi partition with three generators and the Delaunay graph in blue 68
5.4 A configuration with five sensors and the locations of the centroids where the generator points will be moved to in the next iteration when applying the Lloyd algorithm 69
5.5 Visibility-limited Voronoi cells in a nonconvex environment 74
5.6 Environment $\mathcal{Q}$ and the corresponding δ-contraction 75
5.7 Different types of frontiers 76
5.8 Approaching a concave location: the robot positions is shown as a black circle and the red dot indicates the centroid of the VLVC. 80
5.9 Convex scene A with five sensors 82
5.10 Nonconvex scene A with five sensors 82
5.11 Progression of H_{vis} corresponding to Figure 5.10 82
5.12 Nonconvex scene B with four sensors 83
5.13 Nonconvex scene B with limited sensing range and a nonuniform density 83
5.14 Nonconvex scene C with five sensors 84
5.15 Progression of H_{vis} corresponding to Figure 5.14 84

6.1 The three stages of the motivational example mission 87

Abstract

This thesis is about cooperation of multiple robots that have a common task they should fulfill, i.e., how multi-robot systems behave in cooperative scenarios. Cooperation is a very important aspect in robotics, because multiple robots can solve a task more quickly or efficiently in many situations. Specific points of interest are, how the effectiveness of the group of robots completing a task can be improved and how the amount of communication and computational requirements can be reduced. The importance of this topic lies in applications like search and rescue scenarios, where time can be a critical factor and a certain robustness and reliability are required. Further the communication can be limited by various factors and operating (multiple) robots can be a highly complicated task.

A typical search and rescue mission as considered in this thesis begins with the deployment of the robot team in an unknown or partly known environment. The team can be heterogeneous in the sense that it consists of pairs of air and ground robots that assist each other. The air vehicle – abbreviated as UAV – stays within vision range of the ground vehicle or UGV. Therefrom, it provides sensing information with a camera or similar sensor that might not be available to the UGV due to distance, perspective or occlusion. A new approach to fully use the available movement range is presented and analyzed theoretically and in simulations. The UAV moves according to a dynamic coverage algorithm which is combined with a tracking controller to guarantee the visibility limitation is kept.

Since the environment is at least partly unknown, an exploration method is necessary to gather information about the situation and possible targets or areas of interest. Exploring the unknown regions in a short amount of time is solved by approaching points on the frontier between known and unknown territory. To this end, a basic approach for single robot exploration that uses the traveling salesman problem is extended to multi-robot exploration. The coordination, which is a central aspect of the cooperative exploration process, is realized with a pairwise optimization procedure. This new algorithm uses minimum spanning trees for cost estimation and is inspired by one of the many multi-robot coordination methods from the related literature. Again, theoretical and simulated as well as statistical analysis are used as methods to evaluate the approach.

After the exploration is complete, a map of the environment with possible regions of higher importance is known by the robot team. To stay useful and ready for any further events, the robots now switch to a monitoring state where they spread out to cover the area in an optimal manner. The optimality is measured with a criterion that can be derived into a distributed control law. This leads to splitting of the robots into areas of Voronoi cells where each robot has a maximum distance to other robots and can sense any events within its assigned cell. A new variant of these Voronoi cells is introduced. They are limited by visibility and depend on a delta-contraction of the environment, which leads to automatic collision avoidance. The combination of these two aspects leads to a coverage control algorithm that works in nonconvex environments and has advantageous properties compared to related work.

Kurzfassung

Das Thema dieser Dissertation ist die Kooperation in Multi-Roboter Systemen, also wie mehrere Roboter eine gemeinsame Aufgabe lösen können. Kooperation ist ein wichtiger Aspekt in der Robotik, da mehrere Roboter das Potential haben, viele Aufgaben wesentlich schneller oder effektiver zu lösen, als es einem einzelnen, vielleicht komplexeren Roboter möglich wäre. Speziell von Interesse ist im Rahmen dieser Arbeit, inwiefern die Effektivität eines Roboterteams verbessert werden kann und ob eine Reduzierung von Kommunikations- und Rechenaufwand möglich ist. Ein wichtiges Anwendungsgebiet ist durch Such- und Rettungsmissionen gegeben, bei denen Zeit eine kritische Rolle spielen kann und Robustheit sowie Zuverlässigkeit verlangt werden. Des Weiteren kann die Kommunikation in solchen Szenarien durch verschiedene Faktoren eingeschränkt sein und das manuelle Steuern mehrerer mobiler Roboter stellt aufgrund der Komplexität hohe Anforderungen an die Operatoren.

Eine typische Such- und Rettungsmission, wie sie in dieser Arbeit betrachtet wird, beginnt mit dem Einsatz des Roboterteams in unbekanntem oder teilweise bekanntem Gebiet. Das Team kann eine Heterogenität in Form von Luft- und Bodenrobotern aufweisen, die sich gegenseitig unterstützen. Das Luftfahrzeug – kurz UAV – bleibt innerhalb des Sichtbereiches zum Bodenfahrzeug oder UGV. Von dort liefert es Sensorinformationen mit einer Kamera oder einem ähnlichen Sensor. Diese Information ist dem UGV aufgrund von Abstand, Perspektive oder Sichtverdeckungen möglicherweise nicht direkt zugänglich. Um den verfügbaren Bewegungsspielraum maximal auszunutzen, wird ein neuer Ansatz vorgestellt und theoretisch sowie in Simulationen analysiert. Das UAV bewegt sich nach einem dynamischen Coverage–Verfahren in Kombination mit einem Folgeregler, um den Sichtkontakt aufrecht zu erhalten.

Da die Umgebung zumindest teilweise unbekannt ist, wird eine Erforschungsmethode benötigt, um Informationen über die Situation und mögliche Ziele oder Bereiche von Interesse zu identifizieren. Dazu wird ein bestehender Ansatz für die Exploration mit einem Roboter, der auf dem Traveling Salesman Problem aufbaut, auf Multi-Roboter Explorationen erweitert. Die Koordination als zentraler Bestandteil des kooperativen Er-

forschungsprozesses wird mit einer paarweisen Optimierung umgesetzt, bei der minimale Spannbäume als Kostenschätzer verwendet werden. Dieser neue Ansatz wird ebenfalls theoretisch analysiert und in simulierten Experimenten mit relevanten Ansätzen aus der Literatur verglichen.

Nachdem die Exploration beendet ist, liegt eine Karte der Umgebung gegebenenfalls mit Regionen von höherem Interesse vor. Um auf weitere Ereignisse vorbereitet zu sein und die Umgebung sensorisch abzudecken, schaltet das Roboterteam in einen Beobachtungsmodus. In diesem Zustand geht es darum, dass die Roboter sich möglichst optimal verteilen. Die Optimalität wird über eine Gütefunktion definiert, die zu einem verteilten Regelgesetz entwickelt werden kann. Die Roboter teilen sich in Voronoi–Zellen auf, in denen sie für die dort auftretenden Ereignisse zuständig sind. Um einen Einsatz in nichtkonvexen Gebieten zu ermöglichen, wird eine Variation der Voronoi–Zellen eingeführt und mit der Kontraktion der Umgebungsgrenzen verbunden. Dadurch entsteht ein automatisches Hindernisvermeidungsverhalten. Die vorteilhaften Eigenschaften des neuen Algorithmus werden dargelegt und diskutiert.

1 Introduction

This thesis is about mobile robot systems. What is a robot? Why are we developing and building these systems? Why do we want multiple robots to cooperate? What are the specific research goals addressed here? These questions will be answered in this introductory chapter.

As cited many times, the word ’robot’ originates from Czech ’robota’ introduced by Karel Čapek in his play *Rossum’s Universal Robots* in 1921 [124]. It literally means serf labor, figuratively drudgery or hard work and more general work or labor. This gives a first clue as to what a robot might be although chances are high that the understanding of the term has changed since its first introduction. According to this etymology, a robot is a (slave) worker, taking on repetitive or (mechanically) difficult tasks.

The general idea of a self-operating machine is of course much older, by then termed automaton or plural automata. This terminology is closer to automatic control, which is the modern discipline devoted to realization of these original and nowadays much more advanced ideas. But more of that later.

Dating back to Greek mythology humans have been dreaming of machines or control mechanisms that follow a predetermined sequence of operations or respond to instructions. In the *Iliad* [55], ancient poet Homer describes miraculous inventions by Hephaestus, the Greek god of blacksmiths, craftsmen, artisans, sculptors, metals, metallurgy, fire and volcanoes:

> ...for he was fashioning tripods, twenty in all, to stand around the wall of his well-builded hall, and golden wheels had he set beneath the base of each that of themselves they might enter the gathering of the gods at his wish and again return to his house, a wonder to behold.

While the main purpose of these tripods appears to be aesthetic or for entertainment, a new aspect is introduced in Book I of Aristotle’s (384–322 BC) philosophical work *Politics* [5]:

> ...for if every tool could perform its own work when ordered, or by seeing what to do in advance, like the statues of Daedalus in the story, or the tripods of Hephaestus which the poet says 'enter self-moved the company divine,' –if thus shuttles wove and quills played harps of themselves, master-craftsmen would have no need of assistants and masters no need of slaves.

Similar to the robot terminology, Aristotle hints toward use of automata or self-operating tools as slave workers or assistants. And thinking even further, he suggests they would allow abandoning the use of humans as slaves.

A powerful ideal or utopia where every person on this planet has the freedom and support to reach the peak of Maslow's hierarchy of needs [92], i.e., self-actualization, creative expression, realization of ones full potential, may be possible through means of automation. Machines do all the "unattractive" work and take care of the basic needs and supplies for all humans, enabling a more liberated and joyful existence on this planet. Many other societal, cultural and economical factors would of course play an important role for such a development, but these considerations extend the scope of this thesis and shall be left to the experts in this field for now.

1.1 Motivation

Moving from a historical perspective and a general motivation, we will look at more specific aspects as to why robotic systems are a huge and still growing research area. One of the earliest and most obvious motivations to develop an automatism is to overcome physical limitations or alleviate physical labor, i.e., using wind mills for grinding, pumping water or in modern times industrial robots to move heavy parts. Other physical aspects, where humans can be outperformed by specialized robots are speed, precision, perception and mobility. Application examples are numerous and from many different fields like industrial production [154], cleaning [115], security [3] and surveillance or military use [100, 104]. Using robots for production can further be motivated by economical reasons (efficiency, rationalizing), objectivity, safety and reliability (quality management). Robots are also used to protect humans from working with dangerous sources like chemicals or radioactivity [88].

Social contexts like elder care [10], household environments [31], education or entertainment [96] provide many possibilities for robot usage.

Lastly, robots are also being employed in mankind's strive for higher goals like exploring space and other planets [117].

The primary motivating application considered here is the use of mobile robots in search and rescue scenarios or emergency response situations, which will be elaborated in Section 1.3. The aspects most important for this area are mobility, working in dangerous areas, gathering information and taking over repetitive tasks. Beforehand, an in-depth look into the current state of the art in multi-robot systems literature is provided.

1.2 Cooperative Robotics and Multi-Agent Systems

Since this thesis is about multi-robot systems, the multi-agent and cooperative aspects are considered in more detail in this section. The terms multi-agent system (MAS) and multi-robot system will be used interchangeably throughout this thesis as in most other literature, although the focus here is on robotic systems and a MAS can be used for more general types of systems as well. Furthermore, only mobile robots will be considered in the rest of the thesis, i.e., robots that have the ability to move to different locations, since stationary robots do not provide as many interesting or challenging applications for coordination. The following topics are out of the scope:

- coordination of multiple manipulators, articulated arms, or multi-fingered hands, etc.
- human-robot cooperative systems, and user-interface issues that arise with multiple-robot systems.

There are a few distinct **definitions of cooperation** in the robotics literature. Barnes and Gray [7] describe cooperation as

> ...joint collaborative behavior that is directed toward some goal in which there is a common interest or reward.

whereas Cao et al. [16] are a bit more restrictive:

> Given some task specified by a designer, a multiple-robot system displays cooperative behavior if, due to some underlying mechanism, there is an increase in the total utility of the system.

The second quote puts a condition on the behavior of the robots that is related to the performance of the system while the first quote only considers a common interest or reward.

A performance benefit is also one of the most named **motivations** for use of multiple robots. In these cases, the idea is to split a task like exploring an environment or patrolling a given area to do it more efficiently or quickly. There are however several other reasons to use multiple robots instead of a single robot as well. The decision and actual benefits depend on the type of task and task specific requirements. Trying to coordinate a group of autonomous vehicles on the one hand complicates things significantly, but it also offers a remarkable amount of possibilities to tackle new problems or improve already existing solutions. The following list gives an overview of the most common reasons to use multiple robots [8, 16, 151]:

- Tasks may be inherently too complex (or impossible) for a single robot to accomplish.
- Performance benefits can be gained from using multiple robots.
- Building and using several simple robots can be easier, cheaper and more flexible than having a single powerful robot for each separate task.
- Robustness can be increased through data fusion, information sharing and compensating failures of individual units.
- A multi-robot system has a (better) spatial distribution.
- There is the possibility to distribute the computation for computationally complex missions.

To get an impression of the variety and range of possibilities, an overview on previous work in the areas of multi-agent systems and cooperative robotics is given in the following paragraphs. The topics air-ground cooperation, multi-robot exploration and coverage control will be addressed in more detail in the respective Chapters 3, 4 and 5.

One of the early **surveys** on cooperative mobile robotics was done by Cao et al. in 1997 [16], providing an overview over the early formative stages of cooperative robotics and issues such as group architecture, resource conflict, origin of cooperation, learning and geometric problems. Ten years later, Gazi and Fidan [41] as well as Murray [100] reviewed the state of the art of coordination and cooperative behavior of Multi-Agent Systems from a system dynamics and control perspective.

In 2008, Lynne E. Parker, at this time known for her own research in fault tolerant multi robot cooperation [109], published a comprehensive survey on multiple mobile robot systems including the categories architecture, communication, swarm robots, heterogeneity and task allocation among others [110]. One of the most recent surveys on multi-robot coordination can be found in [151].

A perspective with focus on distributed control and obtaining global behavior from local interaction was taken by Martínez et al. [91] and later Antonelli [4], covering topics like consensus, formation and coverage.

The multitude of publications in this huge field of research covers more articles than would be possible here, so for any further references the reader is referred to the surveys mentioned above.

Simulations are an important tool for most robotics researchers, since the hardware can be expensive and time-consuming to set up. A previous stage of simulation studies is therefore highly useful in many cases. For this purpose, several simulation tools have been developed. The most widespread tools are Gazebo [70], with close support and integration of ROS [118], V-REP [125] and the Player-Stage project [43]. These tools have different advantages and disadvantages and the choice ultimately depends on the users requirements and abilities. Since simulation studies in this thesis are done in a specifically programmed environment, the publicly available tools are not as important hereafter.

When it comes to **methods** and mathematical tools applied to MAS and Cooperative Robotics there is a large variety present in the literature, reflecting the different disciplines and backgrounds involved. Examples include Control Lyapunov functions [103], game theory [80, 133], information theoretic approaches [17, 62], linear temporal logic [68], integer programming [93], gradient-based optimization [24] and genetic algorithms [143].

A popular basic problem in the multi-agent literature is the **consensus** or rendezvous. In it's simplest case, the states of a number of homogeneous agents are supposed to converge to the same value [107, 121]. Stability can be proven depending on the structure of the communication network between the agents. Possible extensions include switching topology and time delays [108], event-based communication [134] or restriction to relative measurements [50]. A more generalized problem formulation requires synchronization of the trajectories of multiple agents' states instead of convergence to a fixed value [64, 82].

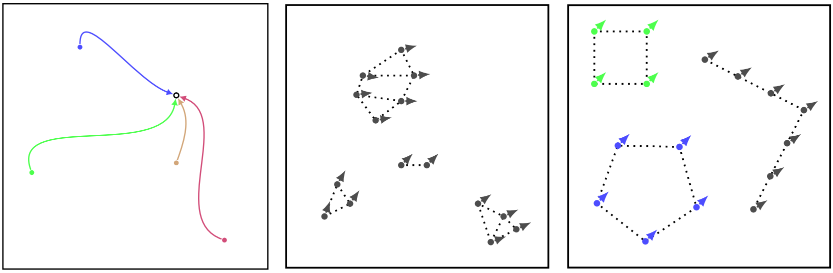
Inspired by flocks of birds and school of fish, Reynolds developed a simulation model for **flocking** [123] – a swarm behavior that is based on three rules:

1. Flock Centering: attempt to stay close to nearby flockmates;
2. Collision Avoidance: avoid collisions with nearby flockmates;
3. Velocity Matching: attempt to match velocity with nearby flockmates.

These local rules lead to a collective, global behavior of the whole system which has drawn interest of other authors [60, 106], leading to more theoretical understanding.

Formation control is another field of research that has led to further research and insights into multi-robot coordination. Tanner et al. [142] investigate the stability properties of mobile agent formations based on leader-follower structures. In [61], the issue of connectedness is of the main concern and the authors are able to guarantee that the graph representing the communication stays connected. A sketch of the consensus, flocking and formation behaviors is shown in Figure 1.1.

When it comes to cleaning, lawn mowing, snow removal or demining, robots will play more and more important roles in the future. The algorithms and theory behind these applications, i.e., finding a path for a robot to pass over all points in a given environment, is called **coverage path planning**. In an early survey [22], Choset characterizes existing approaches into heuristic and provably complete algorithms. They often use cellular decompositions to break down the problem into smaller tasks. A more recent survey [38] provides a good summary on many publications in the field up to 2013. Variations of the basic problem include coverage of unknown environments [42], minimization of repeated coverage when



(a) Consensus **(b)** Flocking **(c)** Formations

Figure 1.1: Different examples for multi-agent control - part 1

using multiple robots [119] or finding the optimal number of robots for a given time limit. A way to look at MR coverage path planning is the multiple watchmen routes problem, where a fixed set of locations to cover all area is computed in a previous step and the second step is to find paths for a number of watchmen (or robots) to visit all locations. This problem has been approached by Faigl [32] with a self-organizing map based adaptation procedure. Focusing on aerial robots and **surveillance**, the authors in [140] and [1] developed cooperative path planning techniques, taking into account limited sensor range and communication and even heterogeneity in sensing and motion capabilities in the second case.

Closely related is the task of **patrolling**, as in [111], the authors try to design optimal multi-agent trajectories to minimize the time gap between any two visits of the same region.

Searching and capturing a single or multiple evaders in minimum time is the challenge tackled in **pursuit evasion games** [104]. Moors et al. [97] specialized in searching an indoor environment for an intruder, taking into account limitations and uncertainties of the robots sensors. Improving the efficiency of a coordinated search is the main focus in [54], using graphical representations of the physical environment to find potentially adversarial targets.

A general way to describe many robotic information gathering tasks is the feedback loop of **active sensing**. Every new bit of information received by a robots sensor is a measurement that can be used to update the current model of the environment which in turn serves as an input for decisions about where to move next, as shown in Figure 1.2. If a camera is used as a sensor, this procedure is also referred to as active perception in the computer vision community [6] and defined as the

> ...problem of controlling strategies applied to the data acquisition process which will depend on the current state of the data interpretation and the goal or the task of the process.

It is therefore an application of control theory but may include more complex data processing, reasoning and decision making as in typical control applications.

Closely related is the process of **spatial estimation**: To learn a field of interest, without further specifying the type of data, a swarm of self-organized autonomous sensing agents is used in [21]. A recursive estimation procedure with radial basis functions provides the model for the unknown environmental data.

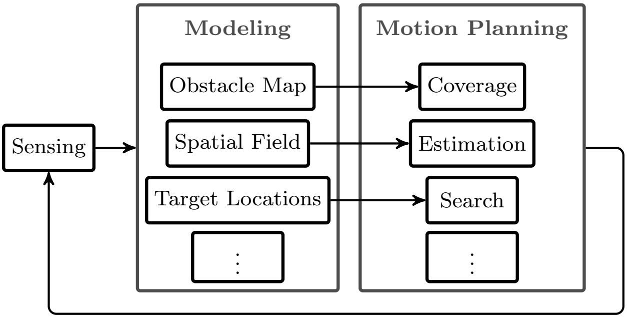


Figure 1.2: Feedback loop for active sensing tasks

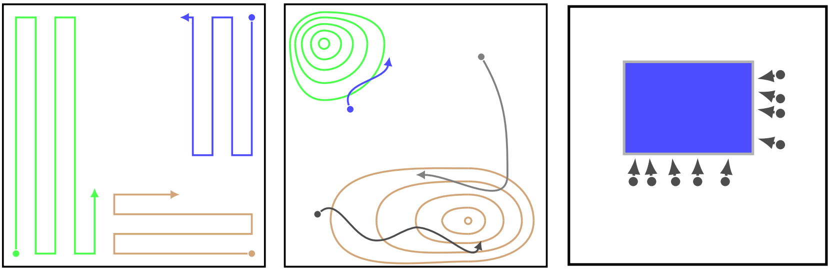
With a strong **information theoretic** background, the authors of the following works provide algorithms and solutions for various active sensing applications: The authors of [136] use nonmyopic planning by taking into account possible observations that can be made in the future for search and rescue and scientific monitoring problems. Choi et al. [20] use mutual information as a metric to reduce uncertainties in the future and apply their ideas to weather forecasting. In a hazardous environment, be it due to radiation, fire or caustic chemicals, collecting informative measurements is even more of a challenge, but with recursive Bayesian filters and again using the gradient of mutual information promising results are possible according to [131]. A distributed information theoretic approach to infer the state of an environment with similar methods as in the previous publication is described in [62]. The authors use a consensus-based algorithm to circumvent the lack of central coordination and to approximate the joint measurement probabilities.

The aforementioned coverage path planning and the process of field estimation are illustrated in Figures 1.3a and 1.3b.

Before these types of algorithms can be applied in a real scenario, some form of **localization** is usually required. If GPS is not available or not accurate enough, as in most indoor locations, another way to localize the mobile platforms is needed. Localization can be achieved through external camera systems like the Vicon[1)] system used by the GRASP Laboratory at the University of Pennsylvania, NASA and ETH Zurich among others. Another possibility is a more low cost camera system as described in [71]. External measurement systems can however be very expensive or time-

---

[1)] Vicon Motion Systems Ltd. - http://www.vicon.com



(a) Coverage path planning **(b)** Field estimation **(c)** Box pushing

Figure 1.3: Different examples for multi-agent control - part 2

consuming to set up and make all experiments depend on the laboratory environment and conditions. A more flexible way to get a relative localization is visual odometry [101], often combined with other sensor data like inertia, gyroscope and wheel encoders.

In a cooperative scenario, there is an additional possibility to use **mutual localization** and calibration. The authors in [120] describe a way for a group of robots to mutually estimate one another's position and uncertainty and they discuss the tradeoffs between sensing and motion control strategies in the context of terrain mapping with multiple robots. Also considering mutual localization, Franchi et al. [37] approach the problem with anonymous relative position measures. A simultaneous calibration method is presented in [145] that not only calculates the relationships between robots but also to the cameras and tools equipped on the mobile agents.

Noteworthy and interesting **application experiments** are published in [19] and [69]. The authors of the first reference demonstrate a swarm of miniature mobile robots that are cooperatively pushing a box to a goal location in a distributed way (cf. Figure 1.3c), even without use of communication. In the second mention, the 'Ikeabots' form a team of heterogeneous robots to complete an assembly task together. With a focus on decentralized cooperative control, Cruz et al. [25] present several application examples and experiments with a real hardware system of multiple ground robots. The tasks include formation control, flocking, goal seeking and target-assessment. For the automatic detection of forest fires, the authors of [94] employ multiple heterogeneous UAVs with infrared and visual cameras as well as fire detectors and use data fusion for cooperative perception. To monitor marine environments, a heterogeneous multi-robot team is presented in [135], assisting scientists to gather information about marine ecosystems and coral reefs in field trials.

1.3 Goals and Contributions

After gaining a broad overview on the field of multi-agent systems and coordinated multi-robot systems, this section gives more specific insights into what the concrete goals and contributions of this thesis are.

The theme of this thesis is to investigate and further develop several aspects of a **complex search and rescue mission**. Imagine the following scenario: After a catastrophe, e.g., a nuclear accident or an earthquake, a team of possibly heterogeneous robots is deployed in the unknown or only partly known environment. There may be victims or other targets of interest still in the area. The team of robots has the task to gather information in the region, find points or areas of importance and then come to a state where they can react to any further events.

The three stages of the mission are shown in Figure 1.4.

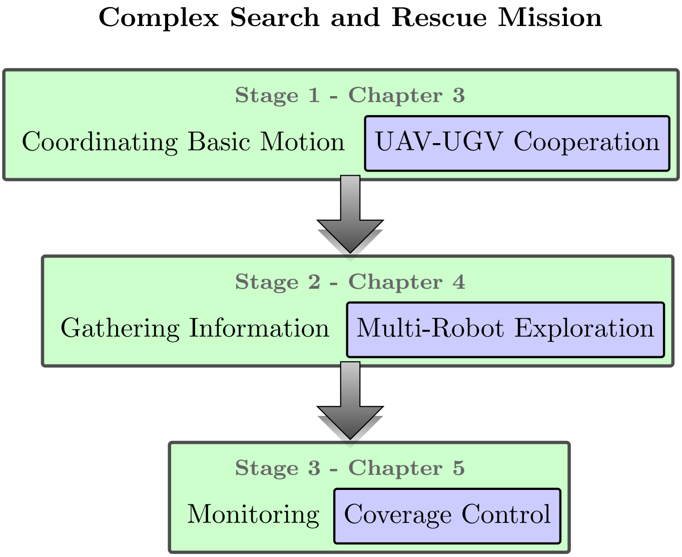


Figure 1.4: Stages of a search and rescue mission as outlined in this thesis

Stage 1 If the team consists of air and ground vehicles, the basic motion control between the UAVs and UGVs is considered, such that the aerial vehicles (e.g., a quadcopter) support the ground vehicles in navigating in difficult areas and provide additional sensory information.

Stage 2 Each respective air-ground team is modeled as a single agent and the area is explored. To this end, a new coordination algorithm is developed and tested.

Stage 3 After the area is explored and all necessary information is gathered, the robots or robot teams switch to a monitoring phase. This phase is characterized by a coverage control algorithm, that positions the robots in an optimal way to react to any new information that might come up.

All three stages require coordination and cooperation between two or more robots. Communication, sensing, path planning and obstacle avoidance and computational complexity have to be considered. The fact that most of the mission takes place in an unknown environment makes a reactive approach necessary, i.e., new information has to be integrated quickly and future actions should incorporate the newly gained knowledge (cf. Figure 1.2). Another important aspect in such situations is that the robot systems have a certain degree of autonomy, especially when multiple possibly heterogeneous vehicles interact or cooperate. Despite the amount of research already done, autonomy is still one of the most unsettled topics. The main goals are to relieve operators from difficult control tasks and make systems more independent when the communication is restricted. These facets make the development and implementation of such a system a highly challenging task.

The **contributions** presented in this thesis are as follows:

- A new approach for motion coordination of air-ground teams is developed and tested in a simulation environment. Exploiting the higher mobility of the UAV and making use of the available visibility region allows for a dynamic and effective way to gather information and support the UGV.

- A new coordination method for Multi-Robot Exploration is presented. The approach is simple, robust and flexible through use of pairwise communication and extensive simulation studies show lower exploration times than other state of the art algorithms.

- A coverage control algorithm for deployment and monitoring with mobile sensors is extended to application in nonconvex environments. The presented method offers low computational effort and communication load compared to other similar approaches.

In addition, the overall system **architecture** is modular and flexible. The stages can be applied individually, depending on the available resources and mission requirements. E.g., if the robot team is homogeneous (consists of only ground or air robots), stage 2 can be initiated directly. If the area is already known beforehand, the robots can move on to stage 3 and monitor the environment for any new occurrences. It is important to note that the stages as illustrated in Figure 1.4 are seen from a development point of view. Stage 1 can also be regarded as a building block for stages 2 and 3. Temporally, the information gathering and monitoring happen one after the other while the basic motion (i.e., stage 1) may be active continuously.

The **methodology** applied to develop the algorithms is always based on an objective function and an optimization, i.e., minimization or maximization of that objective, providing a solid mathematical foundation for the algorithms. The solutions are based on function gradients or heuristics, since the complexity of the problems makes other methods unrealistic on mobile robots with limited processing power and time.

1.4 Thesis Outline

The remainder of the dissertation is structured as follows: The next chapter introduces the most relevant mathematical foundations, including basics in graph theory, computational geometry and dynamical systems.

Chapters 3, 4 and 5 correspond to the three stages of the search and rescue mission outlined in the previous section (cf. Figure 1.4).

The last chapter is a summary and review of the achieved results and gives an outlook on further research or possible future developments.

2 Mathematical Foundations

In this chapter, concepts from several mathematical fields are provided that serve as a basis for the following chapters. The topics of interest are: graph theory, computational geometry (i.e., set theory, Voronoi partitions and the concept of visibility) and dynamical systems. In these areas, that are quite large by themselves, only the aspects relevant for the understanding of this thesis are selected and summarized.

2.1 Graph Theory

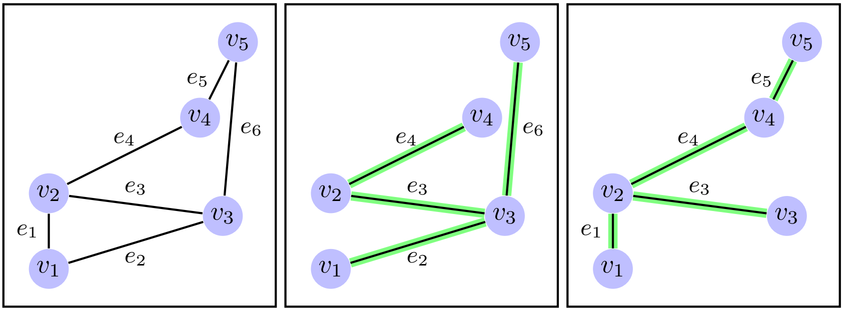
Graphs are a convenient and widespread tool to describe the relationship between multiple vehicles in a multi agent system. A more extensive introduction to the topic can be found in [147] among others.

A **graph** $\mathcal{G}(\mathcal{V}, \mathcal{E})$ consists of a set of **vertices** $\mathcal{V} = \{v_1, ..., v_n\}$ and **edges** $\mathcal{E} = \{e_1, ..., e_m\} \subseteq \mathcal{V} \times \mathcal{V}$ where each edge represents a connection between two distinct vertices. The number of vertices or edges is given by the cardinality of the set, i.e., $n = |\mathcal{V}|$ and $m = |\mathcal{E}|$, respectively. In a multi-robot system, a vertex is usually interpreted as one vehicle and an edge indicates a communication link between two units.

The graph is **weighted**, if there is a weight $w_{ij} \in \mathbb{R}_0^+$ associated with each edge. The set of all weights is referred to as $\mathcal{W}$. If the connection between two vertices depends on the direction, i.e., the weight of edge $(v_i, v_j) \neq (v_j, v_i)$, the graph is called **directed**. Since it is not necessary for this thesis otherwise, we only consider **undirected** graphs where edges are identical in both directions. This concept corresponds to a bidirectional communication in a multi-agent system.

A graph is **complete** or **fully connected** if every pair of distinct vertices $v_i, v_j \in \mathcal{V}$ is connected by an edge.

A **spanning tree** of a weighted graph $\mathcal{G}$ is a subgraph $\mathcal{S}(\mathcal{V}_S, \mathcal{E}_S, \mathcal{W}_S) \subseteq \mathcal{G}(\mathcal{V}, \mathcal{E}, \mathcal{W})$ that connects all vertices and has no cycles (cf. Figure 2.1b). The weight or **cost** of a spanning tree, as of now denoted with $\|S\|$, is the sum of all edge weights in the set of weights $\mathcal{W}_S$: $\|S\| = \sum_{\mathcal{W}_S} w_{ij}$.



(a) An undirected graph **(b)** A spanning tree **(c)** Minimum spanning tree

Figure 2.1: A graph and two spanning trees

Further, a **minimum spanning tree** (MST) is the spanning tree of a graph that has the lowest cost. An example is shown in Figure 2.1c. A MST does not have to be unique for a given graph, i.e., there may be several spanning trees with the same cost. There are various algorithms to find a minimum spanning tree. Popular examples are Prim's algorithm [116] or Kruskal's algorithm [72]. Prim's algorithm proceeds as follows: starting from an arbitrary vertex, the edge with the smallest weight is found and the connected vertex is added to the spanning tree until all vertices are part of the spanning tree. This procedure guarantees that a minimum spanning tree is found and can be completed with a number of operations proportional to $|\mathcal{V}|^2$ or less, depending on implementation.

2.2 Computational Geometry

Sets and set operations are used in this thesis to mathematically describe regions or areas. They provide a notational basis for theoretical concepts and simulations throughout the upcoming chapters of this dissertation. Notations and definitions follow [11].

2.2.1 Sets and Environments

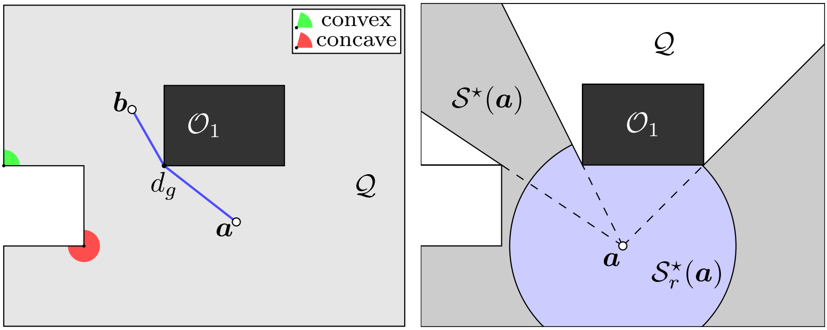
For simplicity and as it is adequate for the considerations in this thesis, the following definitions are given for 2-dimensional real spaces.

A set $\mathcal{Q} \subseteq \mathbb{R}^2$ is called **convex** if and only if the closed line segment $[\boldsymbol{a}, \boldsymbol{b}] = \{\boldsymbol{a} + \lambda(\boldsymbol{b} - \boldsymbol{a}) \mid \lambda \in [0, 1]\}$ is contained in $\mathcal{Q}$ for all $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{Q}$. Otherwise, it is called **nonconvex**.

The boundary and the interior of a set $\mathcal{Q}$ are written as $\partial\mathcal{Q}$ and $\operatorname{int}(\mathcal{Q})$, respectively.

Polygons A **polygon** is a geometric figure that can be described by a finite set of points called **vertices**. The boundary of the polygon is formed by a closed chain of straight line segments connecting these vertices. Sets referred to as a polygons are composed of the boundary and its interior. Using this definition, a polygon always represents a compact set, i.e., the set is closed and bounded. If the boundary does not cross itself the polygon is **simple**.

Environments For further considerations an **environment** consists of a simple boundary polygon $\mathcal{B} \subset \mathbb{R}^2$ and a set of simple polygonal obstacles $\mathcal{O}_1, \ldots, \mathcal{O}_m \subset \mathcal{B}$. The environment is then defined as $\mathcal{Q} = \mathcal{B} \setminus \cup_{i=1}^{m} \mathcal{O}_i$. This is a flexible way of describing arbitrary regions with desired accuracy, regulated by the number of vertices. The polygonal obstacles act as holes or areas where, for example, a robot is not allowed. A vertex of $\mathcal{Q}$ is **convex**, if its interior angle is less than or equal to π radians, otherwise the vertex is **concave** (cf. Figure 2.2a).



(a) Environment and geodesic distance **(b)** Visibility sets

Figure 2.2: Environments and visibility

2.2.2 Distance Metrics

The **Euclidean distance** between two points $\boldsymbol{a} = (a_1, a_2)^\mathsf{T}$ and $\boldsymbol{b} = (b_1, b_2)^\mathsf{T}$ in $\mathbb{R}^2$ is given by $d(\boldsymbol{a}, \boldsymbol{b}) = \|\boldsymbol{b} - \boldsymbol{a}\| = \sqrt{(b_1 - a_1)^2 + (b_2 - a_2)^2}$.

Another useful metric in the context of robotics and path planning is the **geodesic distance** $d_g(\boldsymbol{a}, \boldsymbol{b})$, defined as the shortest path between two points $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{Q}$ that is completely contained in the environment $\mathcal{Q}$. An example is illustrated in Figure 2.2a.

2.2.3 Visibility

Since vision-based sensors play an important role for the applications considered in this thesis, the notion of visibility is elaborated here.

Given a set $\mathcal{Q} \subset \mathbb{R}^2$ and a point $\boldsymbol{a} \in \mathcal{Q}$. A point $\boldsymbol{b} \in \mathcal{Q}$ is **visible** from $\boldsymbol{a}$ if the closed line segment $[\boldsymbol{a}, \boldsymbol{b}] = \{\boldsymbol{a} + \lambda(\boldsymbol{b} - \boldsymbol{a}) \mid \lambda \in [0, 1]\}$ is contained in $\mathcal{Q}$, i.e., $[\boldsymbol{a}, \boldsymbol{b}] \subset \mathcal{Q}$. The set of all points $\boldsymbol{b} \in \mathcal{Q}$ visible from $\boldsymbol{a}$ is the **visibility set** with respect to $\boldsymbol{a}$, denoted as $\mathcal{S}^\star(\boldsymbol{a})$. The visibility set is a star-shaped domain and represents exactly the parts of the environment that are visible from the location of interest.

When modeling a sensor, limited visibility range is often of interest. For circular shapes, this can be described as follows: the **r-limited visibility set** $\mathcal{S}_r^\star(\boldsymbol{a})$ is defined by the intersection of the visibility set $\mathcal{S}^\star(\boldsymbol{a})$ and the closed ball with radius r around $\boldsymbol{a}$, i.e., $\mathcal{S}_r^\star(\boldsymbol{a}) = \{\boldsymbol{b} \in \mathcal{S}^\star(\boldsymbol{a}) | \|\boldsymbol{b} - \boldsymbol{a}\| \leq r\}$. Both types of sets are depicted in Figure 2.2b in a nonconvex environment for an exemplary location.

2.3 Dynamical Systems

Modeling and regulation of dynamical behavior is one of the central building blocks for autonomous robotic systems. The following concepts are covered more extensively in [2, 63].

2.3.1 Nonlinear Systems

A **dynamical system** can generally be described by a **state vector** $\boldsymbol{x} \in \mathbb{R}^n$ that contains all variables needed to characterize the system, the **input vector** $\boldsymbol{u} \in \mathbb{R}^m$ and the vector differential equation

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}). \tag{2.1}$$

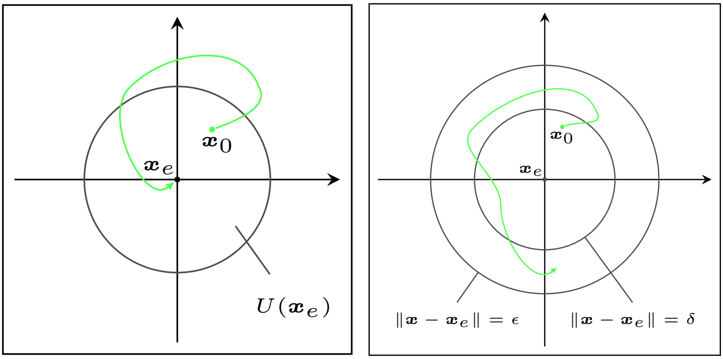
When dealing with dynamical systems, states that do not change over time when the input is zero or constant are of particular interest. Such a state is called **equilibrium point** and defined as a state $\boldsymbol{x}_e$ where $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}_e, \mathbf{0}) = \mathbf{0}$. Solving the equation $\boldsymbol{f}(\boldsymbol{x}_e, \mathbf{0}) = \mathbf{0}$ to compute equilibrium points can be difficult since it is implicit. A single solution, several isolated solutions, a continuum or no solutions are possible.

An equilibrium can have different characteristics:

Definition 2.1. (Attractivity) The **attractivity** is a way to describe the behavior of trajectories in the vicinity of equilibrium points. An equilibrium $\boldsymbol{x}_e$ is locally attractive if there exists a neighborhood $U(\boldsymbol{x}_e)$ such that any initial value $\boldsymbol{x}_0 \in U(\boldsymbol{x}_e)$ leads to a trajectory $\boldsymbol{x}(t)$ that converges to $\boldsymbol{x}_e$ for $t \to \infty$.

As indicated in Figure 2.3a, this notion of attractivity does not restrict the trajectory of the state to diverge arbitrarily far away from the equilibrium before finally converging, unlike the following definition.

Definition 2.2. (Lyapunov stability) An equilibrium $\boldsymbol{x}_e$ is stable in the sense of Lyapunov if for each $\epsilon > 0$ there exists a $\delta > 0$ such that $\|\boldsymbol{x}(t_0) - \boldsymbol{x}_e\| < \delta$ implies $\|\boldsymbol{x}(t) - \boldsymbol{x}_e\| < \epsilon$ for all $t \geq t_0$ (cf. Figure 2.3b). If an equilibrium point is not stable, it is **unstable**. Further, if an equilibrium point $\boldsymbol{x}_e$ is stable and locally attractive, it is **asymptotically stable**.


$\boldsymbol{x}_0$
$\boldsymbol{x}_e$
$U(\boldsymbol{x}_e)$
$\boldsymbol{x}_0$
$\boldsymbol{x}_e$
$\|\boldsymbol{x} - \boldsymbol{x}_e\| = \epsilon$
$\|\boldsymbol{x} - \boldsymbol{x}_e\| = \delta$

(a) Attractivity of an equilibrium point x_e

(b) Stability in the sense of Lyapunov

Figure 2.3: Attractivity and Lyapunov stability, resembling [2]

To analyze the stability of dynamical systems given in the form of (2.1), Aleksandr M. Lyapunov developed a method based on the idea of energy available in a system [86]. The energy level is formulated as a function and if the energy decreases over time, the system will reach a minimum point.

Theorem 2.1. *(Lyapunov's direct method) Assume the differential equation* $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \mathbf{0})$ *has an equilibrium* $\boldsymbol{x}_e = \mathbf{0}$[1] *and a continuous and unique solution for any initial value in the neighborhood* $U(\mathbf{0}) \in \mathbb{R}^n$. *If a continuously differentiable function* $V : \mathcal{D} \to \mathbb{R}_0^+$ *defined on* $\mathcal{D} \subseteq U(\mathbf{0})$ *exists that satisfies*

- *i)* $V(\mathbf{0}) = 0$,
- *ii)* $V(\boldsymbol{x}) > 0$ *for all* $\boldsymbol{x} \in \mathcal{D} \setminus \{\mathbf{0}\}$,
- *iii)* $\dot{V}(\boldsymbol{x}) \leq 0$ *for all* $\boldsymbol{x} \in \mathcal{D}$,

the equilibrium $\boldsymbol{x}_e = \mathbf{0}$ *is stable in the sense of Lyapunov. The function* $V(\boldsymbol{x})$ *is called* ***Lyapunov function****. If in addition* $\dot{V}(\boldsymbol{x}) < 0$ *for all* $\boldsymbol{x} \in \mathcal{D} \setminus \{\mathbf{0}\}$, *the equilibrium point is asymptotically stable.*

The negative derivative of the Lyapunov function characterizes the energy dissipation in the system. Building upon Lyapunov's method, a more extensive and generalized principle has been developed by LaSalle [76]. It allows convergence analysis of several equilibria or more general regions defined as follows.

Definition 2.3. (Invariant sets) A set $\mathcal{M}$ is said to be **invariant** (**positively invariant**), if all initial values $\boldsymbol{x}(t_0) \in \mathcal{M}$ imply $\boldsymbol{x}(t) \in \mathcal{M}$ for all t (for all $t \geq t_0$).

Intuitively, this means that once a trajectory enters a positively invariant set $\mathcal{M}$ it will never leave it again. Using the concept of invariant sets the next theorem is introduced.

Theorem 2.2. *(LaSalle's invariance principle [63]) Let* $\mathcal{D} \subset \mathbb{R}^n$ *be a compact set that is positively invariant with respect to the system dynamics* (2.1) *with* $\boldsymbol{u} = \mathbf{0}$. *Let* $V : \mathbb{R}^n \to \mathbb{R}$ *be a continuously differentiable function such that* $\dot{V}(\boldsymbol{x}) \leq 0$ *for all* $\boldsymbol{x} \in \mathcal{D}$. *Let* $\mathcal{M}_0$ *be the set of all points in* $\mathcal{D}$ *where* $\dot{V}(\boldsymbol{x}) = 0$. *Let* $\mathcal{M}$ *be the largest invariant set in* $\mathcal{M}_0$. *Then, every solution starting in* $\mathcal{D}$ *approaches* $\mathcal{M}$ *as* $t \to \infty$.

---

[1] This assumption can be made without loss of generality, because any equilibrium can be transformed to the origin of the coordinate system.

The advantage of LaSalle's invariance principle are the reduced requirements on the function $V(\boldsymbol{x})$. A negative gradient suffices to prove convergence to the largest invariant set $\mathcal{M}$ without $V(\boldsymbol{x})$ being a Lyapunov function.

2.3.2 Linear Time Invariant Systems

A special type of systems are linear time invariant or LTI systems that can be described by the matrix equations

$$\dot{\boldsymbol{x}} = \boldsymbol{A}\boldsymbol{x} + \boldsymbol{B}\boldsymbol{u} \tag{2.2}$$

$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{x} + \boldsymbol{D}\boldsymbol{u}. \tag{2.3}$$

The linearity makes analysis and regulation of these systems much easier and research in this field is further advanced. Several isolated equilibria as they occur in nonlinear systems are no longer possible. LTI systems or any other ordinary linear differential equation can also be transformed to the Laplace domain. In that case, they no longer depend on the time t but on the Laplace variable s. The relation between a linear multiple input output system and the **transfer matrix** in the Laplace domain is given by

$$\boldsymbol{G}(s) = \boldsymbol{C}(s\boldsymbol{I}_n - \boldsymbol{A})^{-1}\boldsymbol{B} + \boldsymbol{D} \tag{2.4}$$

with $\boldsymbol{I}_n$ being the n-dimensional identity matrix.

A helpful tool for stability analysis of linear systems or differential equations is the following criterion.

Lemma 2.1. *(Routh-Hurwitz criterion [57, 127]) Given a second order polynomial of the form* $P(s) = s^2 + a_1 s + a_0 = 0$*, if the coefficients satisfy* $a_1, a_0 > 0$*, the roots of the polynomial are in the left half plane.*

This means that any linear system with a characteristic polynomial $P(s)$ as defined above is stable if it satisfies the Routh-Hurwitz criterion. The criterion can also be formulated for higher order systems or in a more general form, as shown in the given references.

3 UAV-UGV Cooperation

As outlined in Section 1.3, the first stage of the considered search and rescue scenario is the motion coordination between aerial and ground vehicles. This chapter gives a more in-depth introduction to the topic with a thorough motivation and literature review. Then, the problem is formulated mathematically, beginning with the robot and sensor models that will be used for this stage. Sections 3.3 and 3.4 introduce the two components of the coordination, namely dynamic coverage and a tracking controller, followed by the description of the complete motion control law. Concluding this chapter, simulation results for the cooperative UAV-UGV motion coordination are shown and discussed.[1)]

3.1 Introduction and Literature Review

A team of heterogeneous robots can be much more flexible and versatile, especially in challenging environments like the ones faced in catastrophe or emergency scenarios. The main idea in this chapter is to use the mobility and sensing capabilities of an aerial vehicle like a quadcopter to assist a ground vehicle. Basically, the UAV acts like an extended (mobile) sensor device that provides additional information through complementary sensors on-board the UAV and the different field of view (cf. Figure 3.1). The motion of the ground vehicle is controlled independently, either by performing a task autonomously or via teleoperation, and a priori unknown for now. By not assuming any knowledge about the ground vehicles intentions, the approach is modular and flexible as to be used in different applications. The UAV can provide information that might not be available to the UGV at all, e.g., detecting negative obstacles, surface irregularities, type of terrain etc., or provides information earlier than the UGV would be able to detect with its own sensors. Similar types of systems have been introduced in [95], [18], [40], [26].

For this to be possible, a **relative localization** between the vehicles is required. Since GPS is only available in outdoor scenarios (cf. [46], [75])

[1)] The main results presented in this chapter are also published in Klodt et al. [66].

Figure 3.1: Example of an air and ground vehicle constellation

and point cloud alignment [35] requires an overlap between the sensor data, we focus on relative localization through direct visibility.

Visibility-based relative localization can be realized in different ways, e.g., by detecting the UAV hovering above the UGV with a camera mounted on the ground robot. Cooperative scenarios with this approach have been proposed in [15], [52]. The reverse detection direction is also possible, as demonstrated in [81] among others. In [130], a visibility-constrained formation approach for a group of ground and aerial vehicles is introduced and solved with a model-predictive control approach. A different vision-based approach is described in [99], where the UAV maps an area of interest based on given waypoints and then guides a UGV through obstacles with the gathered camera data. Another popular method in this area is the use of potential forces, as in [141], where a group of fixed wing UAVs circles the centroid of a ground vehicle formation.

In contrast to existing approaches, we deliberately use the region of visibility and the higher mobility of the UAV to gather additional information. To the best of our knowledge, this idea has not been investigated specifically for UAV-UGV systems. We propose and compare two different control laws for motion of the UAV:

1. An extension of a dynamic coverage approach presented in [58, 59] to double integrator systems with a time-varying region of interest.
2. A virtual point tracking method, demonstrated with a circular motion around the ground vehicle's position.

For both cases, we can guarantee that the visibility constraint is satisfied at all times. To this end, we use a tracking controller introduced in Section 3.4 that can be easily parametrized with well-known methods from control theory. The new combined control law is presented in a general form and different weighting functions to combine the tracking with the coverage strategy are evaluated in Section 3.5.

3.2 Problem Formulation

We consider two types of **vehicles**, namely UGVs and UAVs. Variables associated with a specific type of vehicle are denoted with the indices g and a for ground or aerial, respectively.

For rotorcraft aerial vehicles the following simplifications are made: Assuming small angles, the attitude control for the roll, pitch and yaw angles can be decoupled from the position control, as it runs an order of magnitude faster [114]. In a system of nested feedback loops, we consider the outer loop, which is the position control. It is important to note that this simplification is only valid during low velocity and acceleration flight.[2)] Even though the UAV moves in 3-dimensional space, the flight altitude can be considered decoupled from the planar position given our assumptions. Specifying a constant altitude, both ground and aerial vehicles are moving in a convex and bounded environment $\mathcal{Q} \subset \mathbb{R}^2$ on different height levels. Similar assumptions are common in the multi-robot literature, e.g., [146].

Considering the explanations above, quadrotors and omnidirectional ground vehicles can both be modeled by double integrator dynamics

$$\ddot{\boldsymbol{p}}(t) = \boldsymbol{u}(t), \quad \|\boldsymbol{u}(t)\| \leq u_{max}, \quad \|\dot{\boldsymbol{p}}(t)\| \leq v_{max} \tag{3.1}$$

with $\boldsymbol{p}, \boldsymbol{u} \in \mathbb{R}^2$ denoting robot position and control input and $u_{max}, v_{max} \in \mathbb{R}_+$ denoting input and velocity constraints. To allow for higher mobility

---

[2)] More complex models including the full 6-dimensional pose and aerodynamic effects like blade flapping and vehicle body interference are investigated in [53] among others.

of the UAV compared to the UGV we always assume $v_{a,max} > v_{g,max}$ and $u_{a,max} > u_{g,max}$. If the context is clear, time dependencies of most variables are omitted hereafter.

Moving to a target location $\overline{\boldsymbol{p}} \in \mathcal{Q}$ with the UGV can be realized by a stabilizing controller $\boldsymbol{u}_g = \boldsymbol{f}(\boldsymbol{p}_g, \overline{\boldsymbol{p}})$. The goal location $\overline{\boldsymbol{p}}$ is constant until $\|\boldsymbol{p}_g - \overline{\boldsymbol{p}}\| < \epsilon$, for a small $\epsilon > 0$, and then the next goal location is approached. Therefore, the trajectory of the UGV consists of straight line segments.

The **sensor model** of the UAV is defined by a hill-shaped coverage function $S_a : \mathcal{Q} \times \mathcal{Q} \to \mathbb{R}_+$ for vision-based aerial mapping as proposed in [59]:

$$S_a(\boldsymbol{p}_a, \boldsymbol{q}) = \begin{cases} \frac{M_a}{r^4}(\|\boldsymbol{p}_a - \boldsymbol{q}\|^2 - r^2)^2, & \text{if } \|\boldsymbol{p}_a - \boldsymbol{q}\| \leq r \\ 0, & \text{if } \|\boldsymbol{p}_a - \boldsymbol{q}\| > r. \end{cases} \tag{3.2}$$

This function describes how accurate the UAV senses a point $\boldsymbol{q} \in \mathcal{Q}$ given its current position $\boldsymbol{p}_a$, where the sensing quality will be highest at $\boldsymbol{q} = \boldsymbol{p}_a$ (directly below the UAV) with a peak value of M_a. The accuracy S_a declines with increasing distance between the sensed point $\boldsymbol{q}$ and the UAV at position $\boldsymbol{p}_a$ and is zero outside of the sensory range r. An illustration of this function is shown in Figure 3.2. Note that (3.2) is only an example of a possible sensor formulation that could be replaced by any other differentiable function that models the characteristics of a given sensor.

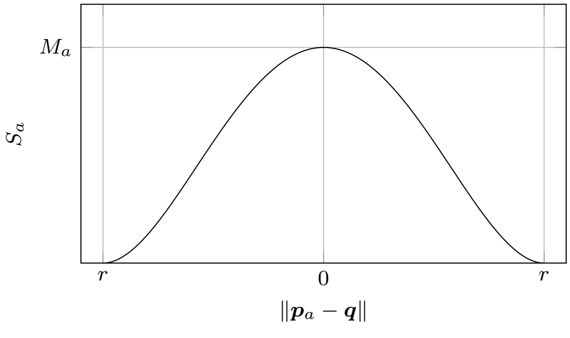


Figure 3.2: Values of the UAV sensory function $S_a(\boldsymbol{p}_a, \boldsymbol{q})$ for different distances of an observed point $\boldsymbol{q}$

The **effective coverage** accumulated at a point $\boldsymbol{q}$ can now be defined by

$$C(\boldsymbol{q}, t) = \int_{0}^{t} S_a(\boldsymbol{p}_a(\tau), \boldsymbol{q}) d\tau \tag{3.3}$$

and can be interpreted as a confidence level about the sensory information attained at that point until time t. To evaluate the current coverage with respect to a desired coverage C^* the following **error function** is used:

$$e(t) = \int_{\mathcal{Q}} h(C^* - C(\boldsymbol{q}, t)) \phi(\boldsymbol{q}, \boldsymbol{p}_g) d\boldsymbol{q}. \tag{3.4}$$

The scalar function $h(x)$ penalizes lack of coverage and can be set to $h(x) = (\max(0, x))^2$ to satisfy the conditions presented in [59]. An additional weighting is achieved by using a density function $\phi : \mathcal{Q} \times \mathbb{R}^2 \rightarrow \mathbb{R}_{\geq 0}$ that is specified more precisely in Section 3.3. As long as the coverage at any point in $\mathcal{Q}$ is below C^* the error $e(t)$ is positive. Otherwise, if $C(\boldsymbol{q}, t) \geq C^*$ for all $\boldsymbol{q} \in \mathcal{Q}$, $e(t)$ is zero since higher coverage values are not penalized.

Finally, the **visibility constraint** to guarantee relative localization is given by

$$\|\boldsymbol{p}_a - \boldsymbol{p}_g\| < r_v \tag{3.5}$$

with $r_v \in \mathbb{R}_+$, $r_v < r$ such that the UGV can always be reliably detected by the UAV.

With these preliminaries, the problem statement can be formulated.

Problem 3.1. (Visibility-based UAV control) Find a stabilizing control law $\boldsymbol{u}_a$ for the UAV with dynamics as in (3.1) that minimizes (3.4) while maintaining the constraint in (3.5) for all t.

3.3 Dynamic Coverage

In the upcoming section a control law for dynamic coverage is revisited and our adaptations and extensions for the current problem are explained.

3.3.1 Original Approach

In [59], a local gradient-type control law for single integrator vehicles $\dot{\boldsymbol{p}} = \boldsymbol{u}_{si}$ is proposed, given by

$$\boldsymbol{u}_{cov} = k\frac{4M_a}{r^4} \int\limits_{\mathcal{Q}_r} h'(C^* - C(\boldsymbol{q}, t)) \cdot (r^2 - \|\boldsymbol{p}_a - \boldsymbol{q}\|^2)(\boldsymbol{q} - \boldsymbol{p}_a)\phi(\boldsymbol{q}, \boldsymbol{p}_g)d\boldsymbol{q} \quad (3.6)$$

with $\mathcal{Q}_r = \{\boldsymbol{q} \in \mathcal{Q} | \|\boldsymbol{p}_a - \boldsymbol{q}\| \leq r\}$, $h'(x) = \frac{\mathrm{d}h(x)}{\mathrm{d}x}$ and $k > 0$.

To understand how the control law works, the individual factors under the integral are analyzed:

- $h'(C^* - C(\boldsymbol{q}, t))$ is a scalar value greater or equal to zero and can be seen as a weighting that increases with higher confidence deficit.
- $(r^2 - \|\boldsymbol{p}_a - \boldsymbol{q}\|^2)$ is a scalar value greater or equal to zero and can be seen as a weighting that emphasizes points that are closer to the vehicle position.
- $(\boldsymbol{q} - \boldsymbol{p}_a)$ is a vector that points from the vehicle position to a point in $\mathcal{Q}_r$.
- $\phi(\boldsymbol{q}, \boldsymbol{p}_g)$ is another weighting factor greater or equal to zero to incorporate additional knowledge about the environment.

Overall, the integral leads to a summation of the weighted vectors pointing from the vehicle position to all other positions $\boldsymbol{q} \in \mathcal{Q}_r$, as illustrated in Figure 3.3. For $\boldsymbol{u}_{si} = \boldsymbol{u}_{cov}$, the single integrator will converge to a state where $\boldsymbol{u}_{cov} = \boldsymbol{0}$. However, it is not guaranteed that the error (3.4) will be zero, due to symmetry or cases where the sensory domain $\mathcal{Q}_r$ is completely covered, i.e., $C(\boldsymbol{q}, t) \geq C^*$ for all $\boldsymbol{q} \in \mathcal{Q}_r$.

To this end, a second control law is introduced. Define a point $\tilde{\boldsymbol{q}}$ that satisfies the following three conditions:

1. $\tilde{\boldsymbol{q}} \in \mathcal{Q}_e$ with $\mathcal{Q}_e = \{\boldsymbol{q} \in \mathcal{Q} | C(\boldsymbol{q}, t) < C^*\}$;
2. $\phi(\tilde{\boldsymbol{q}}, \boldsymbol{p}_g) \neq 0$;
3. $\tilde{\boldsymbol{q}} = \mathrm{argmin}_{\tilde{\boldsymbol{q}} \in \mathcal{Q}_e} \|\boldsymbol{p}_a - \tilde{\boldsymbol{q}}\|$.

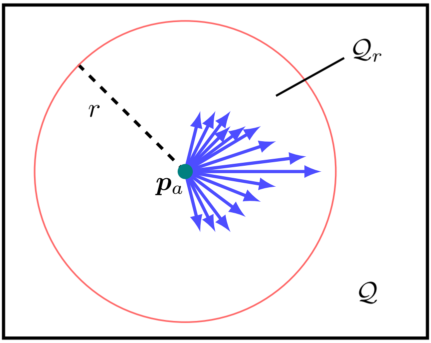


Figure 3.3: Weighted vectors pointing to uncovered areas illustrating the mechanics of (3.6)

These conditions define the closest point to the vehicle position $\boldsymbol{p}_a$ that is not covered and has nonzero density weighting. Approaching $\tilde{\boldsymbol{q}}$ – which is fixed for some time interval – can be realized with a simple linear feedback controller

$$\tilde{\boldsymbol{u}}_{cov} = k(\tilde{\boldsymbol{q}} - \boldsymbol{p}_a). \tag{3.7}$$

If there are multiple points that fulfill all three criteria one can be picked randomly. We can now restate a variant of Theorem III.1 from [58], [59].

Lemma 3.1. *Assuming $\phi(\boldsymbol{q}, \boldsymbol{p}_g)$ is static, a single integrator agent with $\dot{\boldsymbol{p}} = \boldsymbol{u}_{si}$ and a sensor model similar to* (3.2) *and* (3.3) *using the control law*

$$\boldsymbol{u}_{si} = \begin{cases} \boldsymbol{u}_{cov}, & \textit{if } \boldsymbol{u}_{cov} \neq 0 \\ \tilde{\boldsymbol{u}}_{cov}, & \textit{if } \boldsymbol{u}_{cov} = 0 \end{cases} \tag{3.8}$$

drives the error $e(t) \to 0$ as $t \to \infty$.

A proof and stability discussion for this switching control strategy can be found in the references above.

3.3.2 Extensions

The first adaptation that we make is motivated by our visibility constraint. The maximum region the UAV can cover while still respecting (3.5) is a circle with radius $r_o = r_v + r$ centered at $\boldsymbol{p}_g$. Giving higher priority to

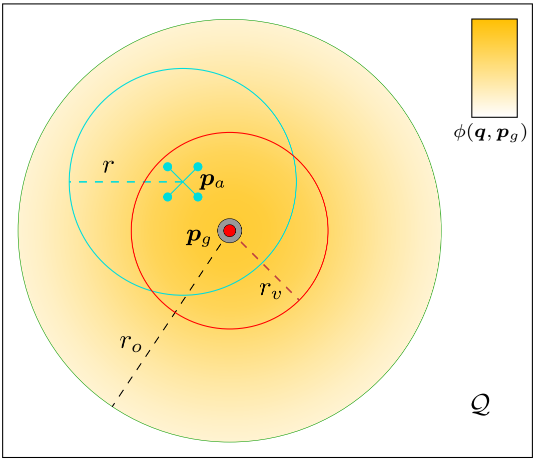


Figure 3.4: The density function as in (3.9)

points closer to the UGV, we define

$$\phi(\boldsymbol{q}, \boldsymbol{p}_g) = \begin{cases} \cos(\frac{\|\boldsymbol{p}_g - \boldsymbol{q}\|}{r_o}\pi) + 1, & \text{if } \|\boldsymbol{p}_g - \boldsymbol{q}\| \leq r_o \\ 0, & \text{if } \|\boldsymbol{p}_g - \boldsymbol{q}\| > r_o. \end{cases} \tag{3.9}$$

As opposed to [59], this definition provides a time-varying density that characterizes the region of importance and limits the area to cover relative to the UGV position. This newly defined density and the circular regions around $\boldsymbol{p}_a$ and $\boldsymbol{p}_g$ are shown in Figure 3.4. The idea of a time-varying density function has also been considered in an adversarial multi-agent scenario in [112].

The second adaptation becomes necessary due to the vehicle dynamics (3.1). We first consider the unconstrained system (i.e., $v_{a,max}$ and $u_{a,max}$ are infinite) and then discuss the case of limited input and velocity. Because (3.6) is a gradient-type control law, it specifies a direction where the error is high and can be reduced by moving in this direction. In a single integrator system, the velocity while following (3.8) will always be $\dot{\boldsymbol{p}} = \boldsymbol{u}_{si}$, i.e., $\boldsymbol{u}_{si}$ can be seen as a desired velocity vector. Specifying a nominal value for the magnitude of the velocity $v_{ref} \in \mathbb{R}_+$, consider the following control error:

$$\boldsymbol{e}_{vel} = \begin{cases} \frac{\boldsymbol{u}_{si}}{\|\boldsymbol{u}_{si}\|} v_{ref} - \dot{\boldsymbol{p}}_a, & \text{if } e(t) \neq 0 \\ -\dot{\boldsymbol{p}}_a, & \text{if } e(t) = 0 \end{cases} \tag{3.10}$$

with $e(t)$ from (3.4). The normalization of the desired velocity provides the advantage that the gain k in (3.6) and (3.7) does not have to be determined and the vehicle moves at a higher velocity even when $\boldsymbol{u}_{cov}$ is very small (a problem that we encountered during simulations). In cases where the error becomes zero, a direction of motion is not defined and the vehicle simply decelerates.

From (3.9), it is obvious that $e(t)$ can never converge to zero permanently as long as the UGV is moving into uncovered regions.

Proposition 3.1. *If $\dot{\boldsymbol{p}}_g(t) = \boldsymbol{0}$ for all $t > t_s$ and any $t_s \geq 0$, a vehicle with dynamics as in* (3.1) *and a sensor model given by* (3.2) *and* (3.3) *using the control law*

$$\boldsymbol{u}_{a,cov} = k_v \boldsymbol{e}_{vel}, \tag{3.11}$$

with $\boldsymbol{e}_{vel}$ from (3.10)*, drives $e(t) \to 0$ as $t \to \infty$ for any $k_v > 0$.*

Proof. As long as the set $\mathcal{Q}_r^* = \{\boldsymbol{q} \in \mathcal{Q}_r | C(\boldsymbol{q}, t) < C^*, \phi(\boldsymbol{q}, \boldsymbol{p}_g) \neq 0\}$ is nonempty, one has $\boldsymbol{u}_{cov} \neq \boldsymbol{0}$ and $\dot{e}(t) < 0$, i.e., the error decreases. If for any reason $\mathcal{Q}_r^* = \emptyset$, one has $\dot{e}(t) = 0$, switching occurs to $\boldsymbol{u}_{si} = \tilde{\boldsymbol{u}}_{cov}$. In this case, inserting (3.10), (3.8), and (3.7) into (3.11) yields

$$\boldsymbol{u}_{a,cov} = \frac{k_v v_{ref}}{\|\tilde{\boldsymbol{q}} - \boldsymbol{p}_a\|}(\tilde{\boldsymbol{q}} - \boldsymbol{p}_a) - k_v \dot{\boldsymbol{p}}_a. \tag{3.12}$$

Without loss of generality, $\tilde{\boldsymbol{q}}$ can be considered a constant reference input and (3.12) provides an asymptotically stable system for any initial state and $k_v, v_{ref} > 0$. This can be shown by formulating the position error $\boldsymbol{e}_p = \tilde{\boldsymbol{q}} - \boldsymbol{p}_a$ and deriving the error dynamics $\ddot{\boldsymbol{e}}_{\boldsymbol{p}} = -\ddot{\boldsymbol{p}}_a = -\boldsymbol{u}_{a,cov} = k_v \dot{\boldsymbol{p}}_a - \frac{k_v v_{ref}}{\|\tilde{\boldsymbol{q}} - \boldsymbol{p}_a\|}\boldsymbol{e}_p$. The differential equation $\ddot{\boldsymbol{e}}_{\boldsymbol{p}} + k_v \dot{\boldsymbol{e}}_{\boldsymbol{p}} + \frac{k_v v_{ref}}{\|\tilde{\boldsymbol{q}} - \boldsymbol{p}_a\|}\boldsymbol{e}_p = 0$ describes a stable system for any point in time due to the positive coefficients according to Lemma 2.1 from Section 2.3.2. Hence, the vehicle position $\boldsymbol{p}_a$ approaches $\tilde{\boldsymbol{q}}$ and will be inside a ball of radius $\epsilon_q < r$ around $\tilde{\boldsymbol{q}}$ at some time $\tilde{t}$. At this point $\mathcal{Q}_r^*$ is nonempty again, $\boldsymbol{u}_{si} = \boldsymbol{u}_{cov}$ and $\dot{e}(t) < 0$ for some time interval. This process repeats until $e(t) = 0$ and convergence is guaranteed since $\mathcal{Q}$ is bounded. □

Note that Proposition 3.1 is also valid for the constrained system with $v_{a,max}, u_{a,max} < \infty$. The maximum velocity can be ignored by choosing $v_{ref} \leq v_{a,max}$. The main difference is caused by the limited control input. In case of unlimited $\boldsymbol{u}_a$ the velocity controller can track the desired velocity with arbitrarily high accuracy by using a high value for k_v. Otherwise the error norm $\|\boldsymbol{e}_{vel}\|$ may be high, depending on the quotient $\frac{v_{ref}}{u_{a,max}}$, which

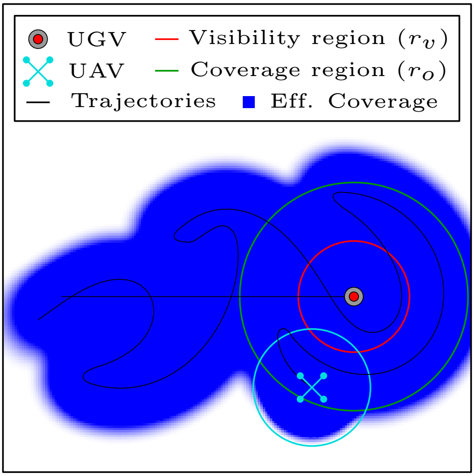


Figure 3.5: Trajectories and covered area for coverage behavior without tracking

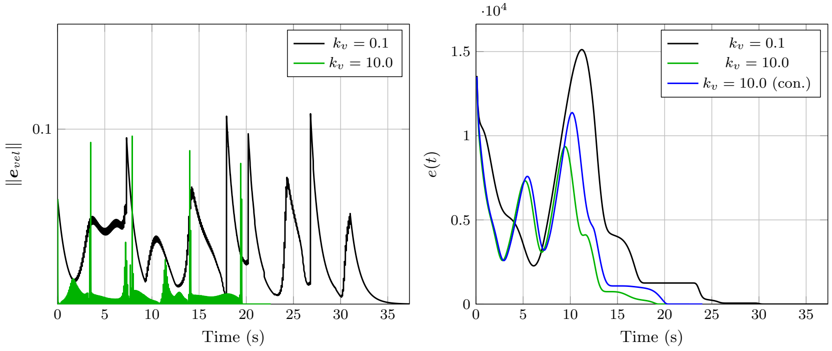
can cause the vehicle to leave the area $\mathcal{Q}$. A safety distance to the boundary becomes necessary depending on the choice of the parameters. This safety distance can be calculated just as the result presented in Proposition 3.2 later in Section 3.5.

3.3.3 Simulation Example

To get a first impression of the coverage behavior, Figure 3.5 shows an example where the UGV is moving along a straight line and the UAV applies the control law given in (3.11). Systems parameters and further details are described in Section 3.6.

Figure 3.6a shows the corresponding velocity error for two different values of k_v. It is obviously much lower for higher k_v and only has a few remaining peaks if the gain is high enough. This supports the assumption that $\boldsymbol{u}_{cov}$ changes slowly and is consistent most of the time with regards to the position of the vehicle, i.e., a slight change in position will change $\boldsymbol{u}_{cov}$ only slightly. The peaks in Figure 3.6a correspond to situations where the direction of $\boldsymbol{u}_{si}$ changes significantly, which happens mostly when switching to the symmetry breaking controller (3.7).

How the coverage error $e(t)$ converges to zero for different k_v is depicted in Figure 3.6b. Again, the higher value of k_v leads to lower error values



(a) Norm of velocity error (3.10) **(b)** Coverage error (3.4)

Figure 3.6: Velocity and coverage error for different control gains corresponding to the trajectories in Figure 3.5

as one would expect. The blue plot shows the error progression for the constrained system ($k_v = 10.0$ (con.)), i.e., when using limited input and velocity for the vehicles. Note that the error can increase as long as the UGV is moving because of the definition of $\phi(\boldsymbol{q}, \boldsymbol{p}_g)$.

As one can see in Figure 3.5, the UAV has left the circle with radius r_v around $\boldsymbol{p}_g$. Therefore, we need an additional motion component that guarantees that the visibility condition is satisfied.

3.4 Tracking Control

In the following, we present a tracking control strategy for the UAV. Note that the visibility constraint (3.5) not only depends on the motion of the UAV, but also on the motion of the UGV. Thus, the UAV needs to follow the UGV to ensure that the visibility condition is satisfied. This problem can be understood as a trajectory tracking problem for the UAV. In this section, we first describe a control law which ensures that the UAV exactly tracks the UGV in case that the coverage task is ignored. Then, in the next section, a combined control law is presented for simultaneous tracking and coverage.

If the UGV is not in motion, the trajectory tracking task is nothing but a regulation of the UAV to a constant point in space. However, if the UGV is in motion, the trajectory is time varying and the solution is more sophisticated. Note that the exact tracking problem is solved if the position error $\|\boldsymbol{p}_\Delta\| = \|\boldsymbol{p}_a - \boldsymbol{p}_g\| = 0$. Considering the dynamics of the position error, it follows that

$$\ddot{\boldsymbol{p}}_\Delta = \ddot{\boldsymbol{p}}_a - \ddot{\boldsymbol{p}}_g = \boldsymbol{u}_a - \boldsymbol{u}_g. \tag{3.13}$$

Therefore, the tracking problem is solved if the error dynamics (3.13) is controlled such that it is stable, since then $\lim_{t\to\infty} \boldsymbol{p}_\Delta(t) = \mathbf{0}$. While $\boldsymbol{u}_g$ is determined for the motion of the UGV, the goal is to design $\boldsymbol{u}_a$ for tracking. A straightforward way is to choose $\boldsymbol{u}_a = \boldsymbol{u}_g + \boldsymbol{u}_{a,tr}$, where $\boldsymbol{u}_{a,tr}$ is designed such that $\ddot{\boldsymbol{p}}_\Delta = \boldsymbol{u}_{a,tr}$ is stable. For example, this can be achieved by a state feedback controller

$$\boldsymbol{u}_{a,tr} = -k_0 \boldsymbol{p}_\Delta - k_1 \dot{\boldsymbol{p}}_\Delta, \tag{3.14}$$

where k_0 and k_1 are chosen such that $\ddot{\boldsymbol{p}}_\Delta + k_1 \dot{\boldsymbol{p}}_\Delta + k_0 \boldsymbol{p}_\Delta = \mathbf{0}$ is stable. However, the controller (3.14) is not only a function of the relative positions $\boldsymbol{p}_\Delta$, but also of the relative velocities $\dot{\boldsymbol{p}}_\Delta$. This implies that the UAV needs information about the velocity of the UGV which is undesirable. To overcome this problem, we use a dynamic control law

$$\dot{\boldsymbol{x}}_c = \boldsymbol{A}_c \boldsymbol{x}_c + \boldsymbol{B}_c \boldsymbol{p}_\Delta \tag{3.15a}$$

$$\boldsymbol{u}_{a,tr} = \boldsymbol{C}_c \boldsymbol{x}_c + \boldsymbol{D}_c \boldsymbol{p}_\Delta \tag{3.15b}$$

which needs only the relative position information $\boldsymbol{p}_\Delta$ to stabilize the tracking error dynamics.

The dynamic controller can be converted to an equivalent transfer matrix, which simplifies the design, i.e., determining the controller parameters. In the Laplace domain, several standard techniques from control theory, e.g., the root locus method can be applied. To this end, using (2.4) yields

$$\boldsymbol{G}_c(s) = \boldsymbol{C}_c (s\boldsymbol{I}_2 - \boldsymbol{A}_c)^{-1} \boldsymbol{B}_c + \boldsymbol{D}_c. \tag{3.16}$$

The structure of the UAV control with the converted dynamic controller is shown in Figure 3.7. Since the two directions in the plane are decoupled, the individual coordinates can be treated like single input output channels making the controller design even more straightforward. For this special case we get the transfer characteristic

$$G_c(s) = \frac{D_c s - A_c D_c + B_c C_c}{s - A_c}, \tag{3.17}$$

assuming $\boldsymbol{A}_c = \operatorname{diag}(A_c), \boldsymbol{B}_c = \operatorname{diag}(B_c)$ and so on.

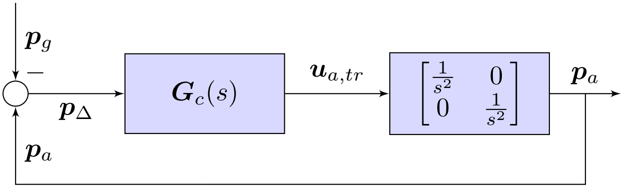


Figure 3.7: Block diagram with the converted dynamic controller for the UAV

Remark 3.1. Relative localization or the information $\boldsymbol{p}_\Delta$ is available for the UAV, as long as the visibility condition (3.5) holds. Once the UAV leaves the visibility radius, the tracking controller (3.15) is useless. Thus, it is necessary to guarantee that the UAV never leaves the visibility radius. In case of failure, an additional backup strategy could still be implemented to reestablish the visual contact.

For the above solution, we have assumed that $\boldsymbol{u}_a = \boldsymbol{u}_g + \boldsymbol{u}_{a,tr}$, meaning that the control input of the UGV is available for the UAV. In general, this is not the case unless this information will be communicated. However, we are interested in solutions which rely solely on the relative position error $\boldsymbol{p}_\Delta$.

For this purpose, it should be noted that the input and velocity of the UGV are bounded by their maximum values $u_{g,max}$ and $v_{g,max}$. Hence, if the UGV is in saturation, it moves with a constant maximum velocity towards its target point. In this case, the UGV can be described by an autonomous vehicle with dynamics $\ddot{\boldsymbol{p}}_g = \boldsymbol{0}$ which implies that $\ddot{\boldsymbol{p}}_\Delta = \boldsymbol{u}_a$ and the information $\boldsymbol{u}_g$ is not needed. Only if the UGV is near its target point, where the system is not in saturation and the velocity is not constant, the information $\boldsymbol{u}_g$ is needed for an exact tracking. However, note that the UAV is more flexible and faster than the UGV such that the tracking error close to a target point is negligible. Moreover, it should be considered that the primary goal is to keep the UAV in a region around the UGV (cf. (3.5)) which does not need an exact tracking and can be ensured without the information $\boldsymbol{u}_g$ in a simple way. We will discuss this issue in the next section.[3)]

[3)] More complex scenarios, e.g., with uneven or difficult terrain, where $\dot{\boldsymbol{p}}_g$ changes frequently are not considered here. Communication of $\boldsymbol{u}_g$ would again be necessary for exact tracking. Approximate tracking will still be possible without $\boldsymbol{u}_g$ available, depending on the ratio of the acceleration and velocity parameters of both vehicles.

Summing up, the motion of the UGV can be described sufficiently accurate by piecewise straight lines with dynamics $\ddot{\boldsymbol{p}}_g = \boldsymbol{0}$, and the dynamics of the tracking error reduce to

$$\ddot{\boldsymbol{p}}_\Delta = \boldsymbol{u}_a. \tag{3.18}$$

Therefore, the controller $\boldsymbol{u}_a = \boldsymbol{u}_{a,tr}$, where $\boldsymbol{u}_{a,tr}$ is given as in (3.15), solves the tracking problem.

3.5 Complete Motion Control

After considering the individual solutions for the tracking problem and dynamic coverage, we can combine the two components to get a complete motion control law that solves the problem posed in Section 3.2.

3.5.1 Combined Coverage and Tracking

While there are different ways to combine the above control laws, we can state the most important aspect of the combined control law in general as follows:

Proposition 3.2. *For a system with dynamics as in* (3.1) *and* $\boldsymbol{u}_{a,tr}$, $\boldsymbol{u}_{a,cov}$ *as in* (3.15) *and* (3.11), *the control law*

$$\boldsymbol{u}_a = \alpha(\|\boldsymbol{p}_\Delta\|)\boldsymbol{u}_{a,tr} + \beta(\|\boldsymbol{p}_\Delta\|)\boldsymbol{u}_{a,cov} \tag{3.19}$$

guarantees $\|\boldsymbol{p}_a(t) - \boldsymbol{p}_g(t)\| < r_v$ *for all* $t \geq 0$ *given the following conditions:*

1. *The initial states satisfy* $\|\boldsymbol{p}_a(0) - \boldsymbol{p}_g(0)\| < r_v$, $\dot{\boldsymbol{p}}_g(0) = \dot{\boldsymbol{p}}_a(0) = \boldsymbol{0}$.
2. $\alpha, \beta : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$ *satisfy* $\alpha = 1, \beta = 0$ *for all* $\|\boldsymbol{p}_a(t) - \boldsymbol{p}_g(t)\| \geq r_s$.
3. *The threshold value* r_s *is given by*

$$\begin{aligned} r_s = r_v - \frac{1}{u_{a,max}} & (v_{g,max}(v_{a,max} + v_{g,max}) \\ & + \frac{1}{2}(v_{a,max}^2 - v_{g,max}^2)). \end{aligned} \tag{3.20}$$

Proof. There is an inner radius $r_s < r_v$ around $\boldsymbol{p}_g$ that determines a distance threshold where only the tracking controller will be active. By considering a worst case scenario for a time t_s when $\|\boldsymbol{p}_\Delta(t_s)\| = r_s$, we can calculate the switching distance r_s such that Proposition 3.2 holds. The worst case scenario is as follows: The UAV moves at $v_{a,max}$ in an arbitrary direction while the UGV moves at $v_{g,max}$ in the opposite direction and $\|\boldsymbol{p}_\Delta(t_s)\| = r_s$ occurs. Now, $\boldsymbol{u}_a = \boldsymbol{u}_{a,tr}$ and the parameters of the tracking controller can be chosen such that the UAV will accelerate with $u_{a,max}$ towards the UGV. The time interval during which $\|\boldsymbol{p}_\Delta\|$ increases can be split into two parts, deceleration to $\|\dot{\boldsymbol{p}}_a\| = 0$ and acceleration to $\|\dot{\boldsymbol{p}}_a\| = v_{g,max}$. Summing up the distances traveled (assuming the UGV still moves away from the UAV) provides (3.20). As soon as $\|\dot{\boldsymbol{p}}_a\| \geq v_{g,max}$ pointing towards the UGV, the distance will not increase any further which completes the proof. □

The choice of $\alpha(\|\boldsymbol{p}_\Delta\|)$ and $\beta(\|\boldsymbol{p}_\Delta\|)$ can have a significant influence on how the error (3.4) evolves. Note that a higher weighting should be given to $\boldsymbol{u}_{a,cov}$ when $\|\boldsymbol{p}_\Delta\|$ is small, meaning that the position tracking is less important when the UAV is closer to the UGV. Several examples of how $\beta(\|\boldsymbol{p}_\Delta\|)$ can be chosen are depicted in Figure 3.8. They correspond to the following function pairs:

$$\alpha_1(\|\boldsymbol{p}_\Delta\|) = \begin{cases} \frac{\|\boldsymbol{p}_\Delta\|}{r_s}, & \text{if } \|\boldsymbol{p}_\Delta\| < r_s \\ 1, & \text{if } \|\boldsymbol{p}_\Delta\| \geq r_s \end{cases} \qquad \beta_1(\|\boldsymbol{p}_\Delta\|) = 1 - \alpha_1(\|\boldsymbol{p}_\Delta\|) \tag{3.21}$$

$$\alpha_2(\|\boldsymbol{p}_\Delta\|) = \begin{cases} 0, & \text{if } \|\boldsymbol{p}_\Delta\| < r_s \\ 1, & \text{if } \|\boldsymbol{p}_\Delta\| \geq r_s \end{cases} \qquad \beta_2(\|\boldsymbol{p}_\Delta\|) = 1 - \alpha_2(\|\boldsymbol{p}_\Delta\|). \tag{3.22}$$

The third option shown is

$$\alpha_3(\|\boldsymbol{p}_\Delta\|) = 1 \qquad \beta_3(\|\boldsymbol{p}_\Delta\|) = \begin{cases} -k_\beta \log(\frac{\|\boldsymbol{p}_\Delta\|}{r_s}), & \text{if } \|\boldsymbol{p}_\Delta\| < r_s \\ 0, & \text{if } \|\boldsymbol{p}_\Delta\| \geq r_s \end{cases} \tag{3.23}$$

with $k_\beta > 0$.4)

In Figure 3.9, the distance norm $\|\boldsymbol{p}_\Delta\|$ for the three different combinations is plotted, considering the same scenario as in Figure 3.5. Obviously, the visibility condition is always satisfied. Note that $\alpha_1(\|\boldsymbol{p}_\Delta\|)$ and $\beta_1(\|\boldsymbol{p}_\Delta\|)$ represent a linear interpolation strategy between $\boldsymbol{u}_{a,tr}$ and $\boldsymbol{u}_{a,cov}$. Although the linear strategy ensures that the UAV stays within

---

4) To avoid numerical issues, $\beta_3(\|\boldsymbol{p}_\Delta\|)$ can be limited to a maximum value.

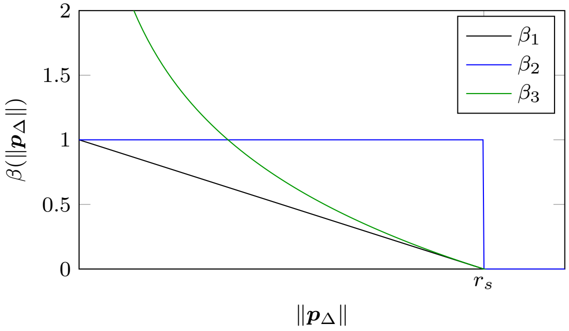


Figure 3.8: Different weighting functions for the combined coverage and tracking control law

the visibility radius r_v, the solution is somewhat conservative since the allowed distance is not fully utilized. From this point of view, a better result is achieved in case of the switching strategy $\alpha_2(\|\boldsymbol{p}_\Delta\|)$, $\beta_2(\|\boldsymbol{p}_\Delta\|)$. However, the controller causes a sliding mode behavior on the threshold r_s which might be undesirable. The choice $\alpha_3(\|\boldsymbol{p}_\Delta\|)$ and $\beta_3(\|\boldsymbol{p}_\Delta\|)$ describes a logarithmic interpolation, which provides a tradeoff between the linear interpolation and the switching function. From Figure 3.9, it can be seen that the UAV moves closer to the visibility boundary compared to the linear interpolation case, but there is no sliding mode as in the switching strategy.

Remark 3.2. Under consideration of Proposition 3.2, other functions for $\alpha(\|\boldsymbol{p}_\Delta\|)$ and $\beta(\|\boldsymbol{p}_\Delta\|)$ are possible. In our simulations, the best results have been achieved with the logarithmic strategy.

3.5.2 Virtual Point Tracking

As an alternative to the combined control strategy presented above, we can also omit the coverage part and simply move the UAV along a specified path around the UGV. For instance, the control law can be modified such that the UAV circles with a certain radius around the UGV while tracking its position.

For this purpose, let us define a virtual point $\boldsymbol{p}_v(t)$ such that

$$\boldsymbol{p}_d(t) = \boldsymbol{p}_g(t) + \boldsymbol{p}_v(t) \quad \text{and} \quad \boldsymbol{p}_d(t) \in \mathcal{Q}. \tag{3.24}$$

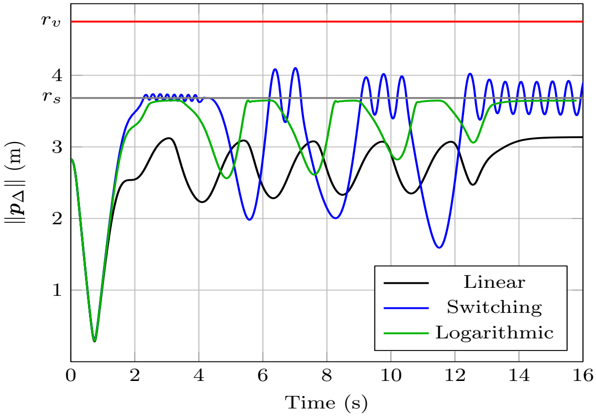


Figure 3.9: UAV-UGV distance for the different weighting functions shown in Figure 3.8

With the help of the virtual point $\boldsymbol{p}_v(t)$, a desired location for the UAV relative to the position of the UGV can be described, where $\|\boldsymbol{p}_v(t)\| < r_v$ must hold to satisfy the visibility constraint (3.5). The vector components of (3.24) are illustrated in Figure 3.10.

The variable $\boldsymbol{p}_d(t)$ can also be interpreted as a trajectory that should be tracked by the UAV, meaning that $\|\boldsymbol{p}_a(t) - \boldsymbol{p}_d(t)\| \to 0$. Thus, as in Section 3.4, we define the tracking error $\boldsymbol{p}_{\Delta,d} = \boldsymbol{p}_a - \boldsymbol{p}_d$ such that the error

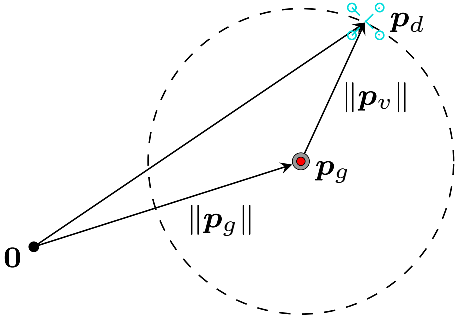


Figure 3.10: Vector components illustrating the virtual point tracking

dynamics are determined by

$$\ddot{\boldsymbol{p}}_{\triangle,d} = \boldsymbol{u}_a - \ddot{\boldsymbol{p}}_d. \tag{3.25}$$

Since we can assume that the dynamics of the UGV are sufficiently accurate described by $\ddot{\boldsymbol{p}}_g = \mathbf{0}$, it follows that $\ddot{\boldsymbol{p}}_d = \ddot{\boldsymbol{p}}_v$. Therefore, (3.25) results in

$$\ddot{\boldsymbol{p}}_{\triangle,d} = \boldsymbol{u}_a - \ddot{\boldsymbol{p}}_v, \tag{3.26}$$

and defining $\boldsymbol{u}_a = \ddot{\boldsymbol{p}}_v + \boldsymbol{u}_{a,tr}$, we get eventually

$$\ddot{\boldsymbol{p}}_{\triangle,d} = \boldsymbol{u}_{a,tr}. \tag{3.27}$$

Obviously, the tracking error (3.27) is described by simple double integrator dynamics which, as before, can be stabilized using the controller (3.15) where it is just necessary to replace $\ddot{\boldsymbol{p}}_\triangle$ by $\ddot{\boldsymbol{p}}_{\triangle,d}$. Thus, provided that the dynamics of the virtual point $\boldsymbol{p}_v$ is twice differentiable with respect to time, a trajectory tracking controller is given by

$$\boldsymbol{u}_a = \ddot{\boldsymbol{p}}_v + \boldsymbol{u}_{a,tr}. \tag{3.28}$$

As an example, assume that the UAV moves in a circle around the UGV, then the virtual point $\boldsymbol{p}_v(t)$ can be specified as

$$\boldsymbol{p}_v(t) = \begin{pmatrix} r_p \cos(\omega_p t) \\ r_p \sin(\omega_p t) \end{pmatrix}, \tag{3.29}$$

where r_p and ω_p are the radius and the frequency of the circular motion, respectively. Applying the same consideration as in Proposition 3.2, choosing $r_p \leq r_s$ guarantees that the visibility condition will be maintained. To satisfy $\boldsymbol{p}_d(t) \in \mathcal{Q}$, i.e., the condition that the virtual point is always inside the bounded environment, $\boldsymbol{p}_d$ is simply projected to the closest position in $\mathcal{Q}$ if $\boldsymbol{p}_d \notin \mathcal{Q}$.

Remark 3.3. In case of a constrained system, the parameters of the virtual point motion should be chosen such that the trajectory can be followed by the vehicle without significant tracking error.

3.6 Simulation Results

In the following, the combined control law and the virtual point tracking method are both illustrated by examples. We implemented the dynamics and controllers in C++ and used the following vehicle parameters: $v_{a,max} = 0.6\frac{\mathrm{m}}{\mathrm{s}}, u_{a,max} = 0.3\frac{\mathrm{m}}{\mathrm{s}^2}, v_{g,max} = 0.2\frac{\mathrm{m}}{\mathrm{s}}$ and $u_{g,max} = 0.1\frac{\mathrm{m}}{\mathrm{s}^2}$.

The sensor characteristics of the UAV are given by $M_a = 1.0$, $r = 5\,\mathrm{m}$, $r_v = 4.75\,\mathrm{m}$ and a desired coverage of $C^* = 2.0$. With these values the threshold for the weighting controller is $r_s \approx 3.683\,\mathrm{m}$. We chose a relatively low velocity and acceleration for the UAV to satisfy the assumption of double integrator dynamics.

The controller parameters for $\boldsymbol{u}_{a,tr}$ in (3.15) are chosen as $\boldsymbol{A}_c = \mathrm{diag}(-3)$, $\boldsymbol{B}_c = \mathrm{diag}(-3)$, $\boldsymbol{C}_c = \mathrm{diag}(-\frac{8}{3})$, and $\boldsymbol{D}_c = \mathrm{diag}(-3)$ such that all closed-loop poles are at -1 in an unconstrained system. According to Remark 3.2, we use the logarithmic weighting function (3.23) for the final experiments with $k_\beta = 20$. Finally, a circular motion is specified by the virtual point as in (3.29) with $r_p = r_s$ and $\omega_p = 1\frac{\mathrm{rad}}{\mathrm{s}}$.

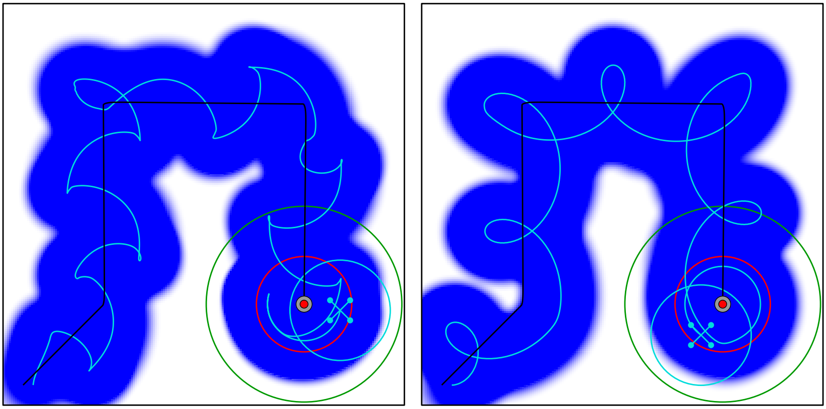
Two different experiments are displayed in Figure 3.11 and Figure 3.12, respectively. The environment is a $40\,\mathrm{m}\times 40\,\mathrm{m}$ square region in both cases. In the first scenario, the UGV travels through completely uncovered space for all t. Therefore, the error (Figure 3.11c) is relatively high, but can be kept lower by the dynamic coverage combined with tracking most of the time compared to the circular motion.

This difference is even more significant if the UGV travels through regions that are partly covered, which is clearly recognizable in the second scenario. Especially in the beginning, when the UGV moves close to already covered areas (between $t = 0$ and $t = 40\,\mathrm{s}$), the error $e(t)$ is much lower and more consistent when using the coverage strategy (cf. Figure 3.12c). This result is not unexpected, because the dynamic coverage approach takes the already covered area and density function into account, but it is also the more complex control strategy.

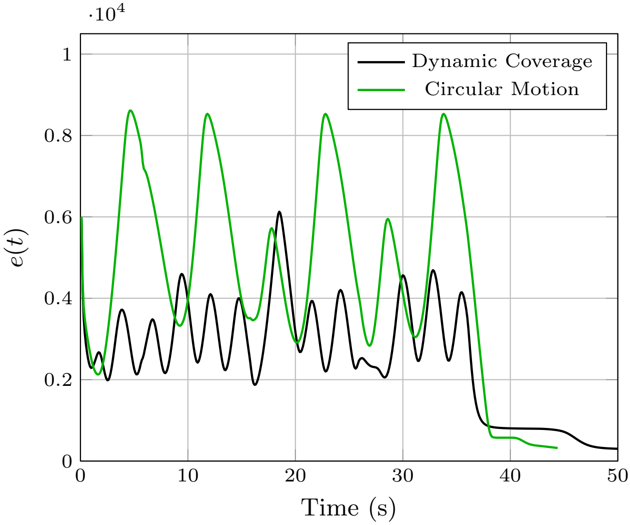
3.7 Discussion

The goal of the work presented in this chapter was motion coordination between aerial and ground vehicles for challenging scenarios. An approach centered on the motion control of the UAV was chosen to exploit the higher mobility and not make too many assumptions about the higher level task performed by the UGV. The cooperation between both vehicles allows for more information to be gathered compared to using a single robot, effectively increasing the sensor range and accuracy during the data collection.

Two strategies were presented that bring along different advantages and disadvantages:

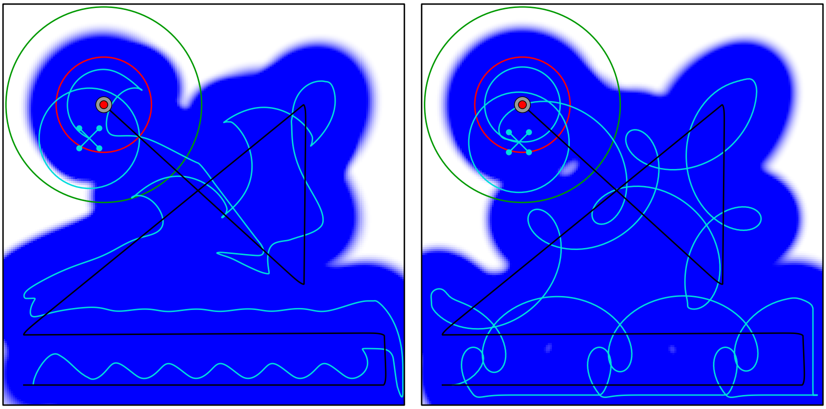


(a) Dynamic coverage trajectories **(b)** Circular motion trajectories

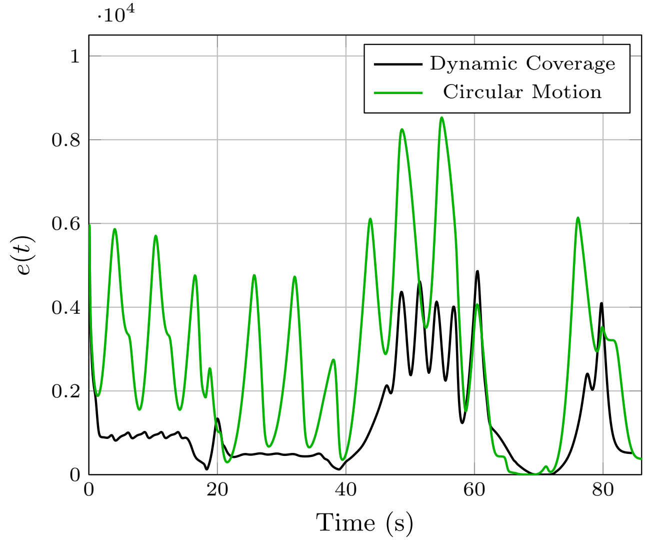


(c) Comparison of $e(t)$

Figure 3.11: Simulation results for scenario 1



(a) Dynamic coverage trajectories **(b)** Circular motion trajectories



(c) Comparison of $e(t)$

Figure 3.12: Simulation results for scenario 2

- The **circular motion** approach is simpler to implement and not as demanding computationally, because it does not require the UAV to keep a map of the environment and evaluation of the gradient or finding the next point to cover as in (3.6) and (3.7), respectively. Therefore, this strategy should be chosen if the processing power of the UAV is an issue or the processing can not be outsourced to an external computer or the UGV via communication.

- The **dynamic coverage** strategy combined with the tracking should be used if possible as it offers the better coverage (as far as our experiments showed) and provides opportunities for even more sophisticated assistance, e.g., by taking additional information given by the UGV about it's goals into account.

- The required visibility contact is guaranteed at all times in both cases.

One UAV-UGV team can be treated as a single entity, since they stay within a defined range of each other and share their sensory information. This is an important aspect that is utilized in the upcoming chapter.

4 Coordination in Multi-Robot Exploration

In this chapter, a specific higher level task mentioned previously is considered in detail. When looking back at the complex search and rescue scenario introduced in Chapter 1, one of the most important steps is the exploration of the unknown environment, i.e., finding out more about the current situation or finding victims to be rescued as quickly as possible. Doing so with multiple robots in a coordinated way poses challenges that are investigated and addressed here.

First, the topic of multi-robot exploration (MRE) is introduced, followed by a literature review and a summary of the contributions presented here. The problem statement starts with a simplification of the robot and sensor models used in the previous chapter to accommodate to the complexity in this new stage. Then the cost function for the exploration problem is presented. A new optimization procedure is proposed and analyzed theoretically in Section 4.3. Simulated evaluation and comparison in an exploration framework complete the analysis that is again followed by a discussion.[1]

4.1 Introduction

The elementary strategy in exploration is the **frontier-based approach** [150], i.e., determining the boundary between known and unknown space and moving towards it until the entire environment is explored. This process is illustrated in Figure 4.1.

There are many different strategies on how to move the robot towards the unexplored regions. The selection of a suitable target location in [150] is solely based on the distance to the closest point on the frontier. This measure can be augmented with different utility functions like expected information gain and localization quality [89]. In 2011, Kulich et al. presented a new strategy for goal selection for a single robot that is based on

---

[1] The main results presented in this chapter are also published in Klodt et al. [67].

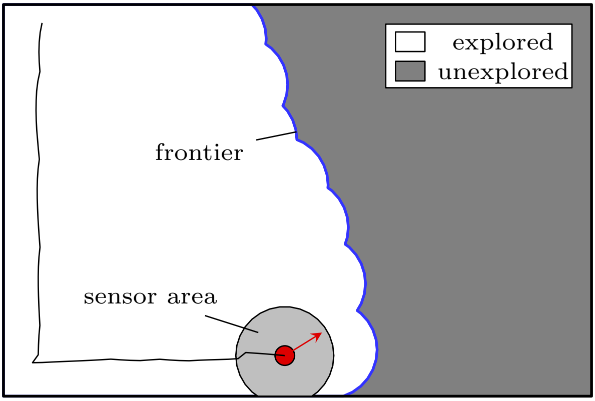


Figure 4.1: Frontier-based exploration with a single robot

repeatedly solving the Traveling Salesman Problem (TSP) and approaching the first target in the open TSP tour [73]. The open TSP tour is defined as the shortest path visiting all available locations without returning to the starting position. It can be seen as a more predictive method and significantly outperforms a greedy goal selection strategy in several scenarios. The advantage is explained through consideration of all future target locations as depicted in Figure 4.2.

In recent years, the focus in robotics research shifted more and more towards multi-robot systems and multi-robot exploration. As already mentioned in the first chapter, employing multiple robots bears the potential to reduce the overall exploration time, increase accuracy and robustness to failures. The reduction in exploration time is strongly dependent on effective coordination of the group, whereas additional robustness emerges from a decentralized system structure where failures of an individual unit can be compensated. Further, centralized coordination can be inefficient in terms of communication and computation requirements when the central controller becomes the bottleneck of the system [8]. These arguments provide a good motivation to focus on a decentralized or even fully distributed approach.

Aside from minimizing the exploration time, which is our focus here, there are also other types of team objectives that can be considered in the multi-robot exploration context. Depending on the type of application, many aspects can be imagined, e.g., minimizing usage of resources

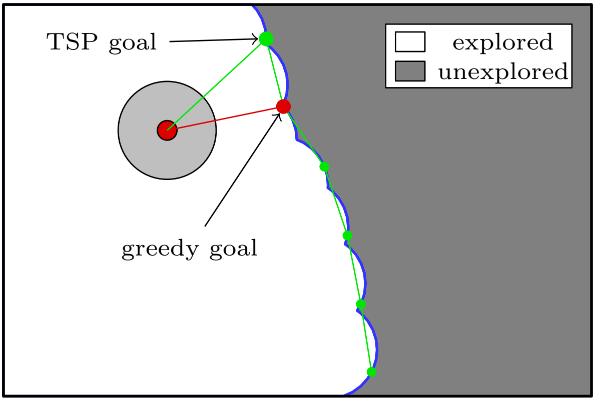


Figure 4.2: Comparison of greedy and TSP-based goal selection

like energy, computation or communication or having a system with high robustness.

4.1.1 Literature Review

There is a large amount of literature on coordination and task allocation methods in multi-robot systems that cannot be fully covered here. We focus on the most relevant and representative approaches from our point of view. For a more complete survey see [151] for example.

As stated in [12], the key problem regarding effective coordination is the allocation of target points such that individual robots explore different regions of the environment. A simplified variant of this problem is the assignment of a single task to each robot, which has been widely studied in other fields like operations research. It can be solved optimally in polynomial time with the Hungarian method which has been applied to multi-robot exploration by the authors of [149]. Assuming structured room environments, the authors first segment the explored area based on a Voronoi graph. The resulting segments are then assigned to different robots.

Assigning multiple targets to each robot is sometimes approached as a partitioning problem in the multi-robot exploration literature. Examples include the use of k-means clustering [138], [33] or Voronoi cells [148], [48], [49]. The partitioning is used as a heuristic to distribute the robots to different regions of the environment. A common problem to these partitioning methods is an uneven assignment of work, as they do not consider the actual path the robot will have to travel. Therefore the Voronoi partition and even k-means clustering can lead to clearly suboptimal decompositions, including the possibility of empty cells or clusters. In [33], the issue is subsequently addressed by applying a second assignment step that sets goal locations for all unemployed robots.

A different means to assign sets of tasks to multiple robots are market-based approaches. A lot of research has been focused on coordination with market-based mechanisms and the results up to 2006 are summarized in [27]. In this context the multi-robot task allocation (MRTA) is usually formulated as an optimal assignment problem, which is typically NP-hard as it contains difficult subproblems from combinatorial optimization [44]. Therefore most approaches focus on finding a (presumably suboptimal) solution that reliably produces good solutions.

One of the important paradigms for decentralized task allocation was introduced as the contract net protocol (CNP) in [137] and many articles present algorithms as variants or extensions of the CNP. The basic idea is that agents can announce tasks similar to an auction and other agents can formulate bids on these tasks. Examples are [45], [129] and [155]. Common to these approaches is that they consider individually rational agents that calculate bids based on their own utilities (i.e., what is my gain for adding or removing a certain task to/from my task set). While [45] only allows task swaps, the authors of [129] present theoretical results on four different types of contracts (i.e., the ways to exchange tasks) with the interesting conclusion that none of the four contract types alone is sufficient to reach the optimal allocation. An application example is the work of Zlot et al. [155], where each robot has a tour of goals and tries to sell or buy target points by evaluating expected profit in adding that goal to its current tour. The main disadvantage of using self-interested agents with marginal cost based contracting is that it corresponds to the MinSum objective from an optimization point of view and such a system is not suitable to minimize exploration time or facilitate balanced use of all robots (cf. [98]). Realizing this issue, the approach in [79], another extension to CNP, uses an equity coefficient in the robots utility functions to balance the workload between them. Agents with higher workload also gain communication priority when it comes to giving away tasks.

A more profound perspective is provided by Lagoudakis et al. in [74]. They present three different team objectives for the multi-robot routing (MRR) problem and derive bidding rules for the agents directly from there. Since computation of path costs for the individual robots is already NP-hard, they propose use of minimum spanning trees (MST) or a polynomial time TSP heuristic as cost approximations. Even with the approximations they can prove upper bounds on the performance ratio of the algorithms.

Since the way auctions are designed can severely limit the algorithms capabilities of reaching a good allocation, several approaches have been made to find more complex or sophisticated ways to exchange tasks. The authors from [8] employ combinatorial auctions for exploration in partially unknown terrain, taking synergies into account that arise from points closer together. In [152] a new type of contract to exchange multiple targets is defined and shown to improve the initial allocation given by other algorithms, based on the MinSum objective. Combining the ideas of clustering and auctions (cluster first, then allocate), a comparison of two cluster formation techniques is presented in [51] with the conclusion that single-linkage clustering can provide minor improvements over k-means.

4.1.2 Contribution

Originally inspired by issues that are present in MRE when using partitioning methods (uneven allocation, centralization) we found that allocation techniques exist in the MRR literature to overcome these problems, especially the solutions presented in [74]. However, in our experiments they have shown to be lacking for highly dynamic applications with frequent replanning[2)] like the exploration scenarios we are considering. It is important to note, that most of the algorithms from MRTA or MRR have only been applied and tested in static scenarios with the intention to find the best allocation for a given (and a priori known) situation. Additional factors come into play when it comes to dynamic exploration. We presume that a certain consistency between solutions of consecutive assignment iterations is advantageous, and therefore sticking to a solution closer to the previous one can prove beneficial. Reallocation of targets based on the previous solution has already been applied in [155] among others but with a different optimization objective and communication scheme.

With these arguments in mind, we designed a new allocation algorithm that combines the advantages of above mentioned approaches while posing fewer requirements on the communication structure. Specifically, we are using the MinMax criterion to minimize the overall exploration time and, similar to [74], use minimum spanning trees to approximate the robot path

---

[2)] The necessity of frequent replanning has already been investigated in [33] and their results also match with our own experience.

costs. Always starting from the previous allocation, the robots execute a pairwise optimization procedure, minimizing the global objective through local interactions. Compared to many popular auction strategies this is a reduction in communication requirements since it is only necessary to have a stable connection with one agent for an optimization step to proceed, instead of waiting for the bids from all other robots (usually with a timeout to increase robustness towards failed robots or robots not within range).

Our contributions summarized in short:

- We combine insights from MRE and MRR/MRTA to design a new assignment algorithm...
- ...that is adapted for highly dynamic scenarios like exploration...
- ...and has reduced requirements on the communication structure of the system.
- We provide a theoretical analysis of convergence properties.
- We provide a statistical evaluation that compares the proposed algorithm with state of the art approaches.

4.2 Problem Statement

This section introduces the general set-up and the cost function for optimization.

4.2.1 Setting and Assumptions

To accommodate to the increased complexity and focus on the task of exploration, the models used in Chapter 3 are simplified as follows:

1. The multi-robot team is no longer assumed to be heterogeneous. Instead, each UAV-UGV pair is merged into a single **exploration unit** with an increased sensor range r_e (compared to a single robot) to account for the additional information gathered.
2. Within sensor range r_e, an exploration unit can determine if there is free space unless the sensing is blocked by an obstacle.
3. The movement of an exploration unit is modeled by a single integrator.

For simplicity, exploration units are also called robots hereafter. Assuming a group of $n < \infty$ robots, they also carry the following idealized capabilities:

1. A robot can communicate with other robots within a communication range r_c. Those available for direct communication are also called neighbors.

2. The robots can localize in a common coordinate frame, which allows them to exchange map or target information with their neighbors.

A practical realization of the localization assumed here can be found in [36].

The environment considered for exploration consists of a polygonal boundary and a set of polygonal obstacles as defined in Section 2.2.1 and is denoted with $\mathcal{Q}$. Note that the environment is now allowed to be nonconvex in contrast to the previous chapter, which is more fitting for real exploration scenarios. Define the **maneuverable domain** $\mathcal{D} \subset \mathcal{Q}$ as the union of explored and unexplored areas excluding explored parts of obstacles and boundaries. As a reminder, the geodesic distance as defined in Section 2.2.2 between any two points $\boldsymbol{a}, \boldsymbol{b} \in \mathcal{D}$ is denoted with $d_g(\boldsymbol{a}, \boldsymbol{b})$.

Remark 4.1. We include the unexplored regions in the maneuverable domain, i.e., we assume that the space is obstacle free unless proven otherwise. This requires consistent updating of the motion planning as new obstacles are discovered but also leads to a more offensive exploration.

4.2.2 Cost Function

One of the main challenges in assigning equal amounts of work for exploration is to estimate the actual workload for each individual robot. The difficulty of the problem lies in the high uncertainty inherent to the exploration problem, namely the region beyond the exploration frontier. There could be a wall directly behind the frontier or maybe most of the unexplored area awaits to be discovered. Without any additional knowledge or assumptions about the environment, one is usually forced to rely on the length of the frontier (or number of suitable target locations close to the frontier) as an indicator.

Having M_i target points $\boldsymbol{t}_{k_i} \in \mathcal{D}$, $k_i \in \{1, \ldots, M_i\}$, assigned to a robot with index i at position $\boldsymbol{p}_i$, $i \in \{1, \ldots, n\}$, the robots path cost or workload amounts to

$$w_i = d_g(\boldsymbol{p}_i, \boldsymbol{t}_{1_i}) + \sum_{k_i=1}^{M_i-1} d_g(\boldsymbol{t}_{k_i}, \boldsymbol{t}_{k_i+1}), \tag{4.1}$$

where the target points are ordered such that w_i is minimal (i.e., the open TSP tour). This distance measure can be seen as an approximation of the distance a robot will have to travel. Either there is only a small area left to be explored (all within a distance r_e of the target points), then the robots only have to visit the remaining targets (or move close to them) and the exploration is complete. Otherwise the travel distance will be higher as additional target points arise on the updated frontier.

4.2.3 Approximating the Cost Function

To allow for frequent re-computation of workloads, we use an MST approximation (cf. [74]) described in the following. First, define the complete, undirected graph $\mathcal{G}_i(\mathcal{V}_i)$ of robot i with vertices $\mathcal{V}_i = \{\boldsymbol{p}_i, \boldsymbol{t}_{1_i}, ..., \boldsymbol{t}_{M_i}\}$. The weight of an edge between two vertices $\boldsymbol{u}, \boldsymbol{v}$ is given by their distance $d_g(\boldsymbol{u}, \boldsymbol{v})$.

Proposition 4.1 (MST Approximation)**.** *Let $\mathcal{S}_i$ be a minimum spanning tree of the complete, undirected graph $\mathcal{G}_i(\mathcal{V}_i)$, vertices and edge weights defined as above and the workload w_i of a robot as in* (4.1)*. Then, the following observation holds:*

$$\|\mathcal{S}_i\| \leq w_i < 2\|\mathcal{S}_i\|. \tag{4.2}$$

Both inequalities are basic results from TSP research transferred to our problem, a proof can be found in the related literature, e.g., [126]. The MST cost provides a lower and upper bound and therefore a meaningful approximation of w_i. This observation is useful because the MST can be computed deterministically with time complexity $O(|\mathcal{V}_i|^2)$. Other polynomial time TSP heuristics would also be possible, but do not provide any advantages. Therefore, for further considerations we use the MST weight $\|\mathcal{S}_i\|$ as an estimation of the workload of robot i. An example of two MSTs in an exploration scenario is shown in Figure 4.3.

Assuming the robots move consistently and at a constant velocity, the overall exploration time directly corresponds to the longest traveled distance by any one robot. This consideration immediately suggests the performance measure

$$\mathcal{L} = \max(\|\mathcal{S}_1\|, \|\mathcal{S}_2\|, \ldots, \|\mathcal{S}_n\|), \tag{4.3}$$

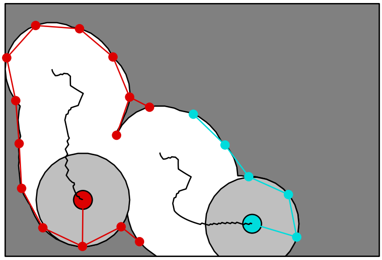


Figure 4.3: Minimum spanning trees between the robot positions and all assigned target points

using the estimated workload as described above. We can now formulate the problem statement.

Problem 4.1. (MRE Coordination) Find a distributed algorithm for partitioning and assignment of target points $\boldsymbol{t}_{k_i}$, that implements the objective

$$\min_{\mathcal{P}} \max(\|\mathcal{S}_1\|, \|\mathcal{S}_2\|, \ldots, \|\mathcal{S}_n\|) \tag{4.4}$$

with $\mathcal{P} = \{\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_n\}$ being a partition of the set of targets and the elements in $\mathcal{P}_i$ are allocated to robot i.

This MinMax objective automatically leads to an equitable partitioning with respect to estimated travel distances. Solving (4.4) combined with an appropriate motion control to visit the assigned targets should yield an exploration strategy with reduced overall exploration time compared to less optimal partitioning or target assignment algorithms.

4.3 Pairwise Optimization

Partly inspired by [30], we propose a pairwise optimization procedure on a discrete set of locations. Following some preliminary considerations, the algorithm is presented and analyzed in the upcoming subsections. The resulting combinatorial optimization problem is basically independent from the discretization, i.e., the generation of target locations, which is introduced in Section 4.4.1. For now, assume that a set of target positions exists.

4.3.1 Searching the Solution Space

The basic idea of the pairwise optimization is to reduce the overall problem from (4.4) into a sequence of subproblems by optimizing the workload in pairs of robots. This reduces communication requirements as only two robots have to communicate at a time and also significantly reduces the complexity.

The subproblem

$$\min_{\mathcal{P}_{ij}} \mathcal{L}_{ij} = \min_{\mathcal{P}_{ij}} \max(\|\mathcal{S}_i\|, \|\mathcal{S}_j\|) \tag{4.5}$$

is a set partitioning problem where the union of target points belonging to i and j (i.e., $\mathcal{P}_i \cup \mathcal{P}_j$) is partitioned into two new subsets. Assigning m targets to robot i is equal to the process of choosing m from a set containing $M_i + M_j$ elements. Including the cases that one robot has all or no targets, the feasible set of solutions $\mathcal{F}$ contains $|\mathcal{F}| = \sum_{m=0}^{M_i+M_j} \binom{M_i+M_j}{m}$ possible partitionings. Using the well-known binomial theorem this result can be reformulated to

$$|\mathcal{F}| = \sum_{m=0}^{M_i+M_j} \binom{M_i + M_j}{m} = 2^{M_i+M_j}, \tag{4.6}$$

meaning the cardinality of the solution set rises exponentially with an increasing number of target points. This is in accordance with the result from [129], where the number of possible assignments for a general system with n agents and M targets is given by n^M. The following example demonstrates how the division into smaller subproblems simplifies the partitioning procedure:

Example 4.1. Consider a team of $n = 4$ robots and a total of $M = 20$ targets, with $M_1 = M_2 = M_3 = M_4 = 5$ targets assigned to each robot initially. The number of possible assignments is $4^{20} \approx 1.1 \cdot 10^{12}$, while each pairwise subproblem only has $2^{10} = 1024$ combinations. In the case of uneven initial allocation, the complexity of a pairwise step increases to $2^{20} \approx 1.05 \cdot 10^6$ at most.

Since each iteration of a pairwise optimization requires an evaluation of the involved agents cost functions, an exhaustive search can still be very costly and would not scale to higher numbers of targets. Therefore the procedure denoted as Algorithm 1 is presented: Starting with an initial allocation and then always based on the previous solution, the **workload**

balancing algorithm performs a local hill descent on the possible partitionings reachable by moving one target to or from the corresponding partner. The robot with the higher cost gives one point to the other by evaluating all possibilities and choosing the one that gives the greatest reduction in pair cost $\mathcal{L}_{ij}$. This step is continued iteratively until no further improvement can be achieved. This simple but effective procedure guarantees convergence to the closest local minimum reachable with the *give one target away* move. Other strategies could be evaluated for cases where there are multiple possibilities to reduce the cost by moving one target, e.g., by taking the sum of costs into account as a secondary objective.

Algorithm 1: Workload Balancing

```
Data: Two point sets $\mathcal{V}_1 = \{\boldsymbol{p}_1, \boldsymbol{t}_{1_1}, ..., \boldsymbol{t}_{M_1}\}$ and
      $\mathcal{V}_2 = \{\boldsymbol{p}_2, \boldsymbol{t}_{1_2}, ..., \boldsymbol{t}_{M_2}\}$
Result: Optimized sets $\mathcal{V}_1^*, \mathcal{V}_2^*$
Compute $\|\mathcal{S}_1\|, \|\mathcal{S}_2\|$ from $\mathcal{V}_1, \mathcal{V}_2$
$\mathcal{L}_{12}^* = \max(\|\mathcal{S}_1\|, \|\mathcal{S}_2\|)$
while $\mathcal{L}_{12}^*$ has improved do
    if $\|\mathcal{S}_1\| > \|\mathcal{S}_2\|$ then
        for $i = 1$ to $M_1$ do
            move $\boldsymbol{t}_{i_1}$ from $\mathcal{V}_1$ to $\mathcal{V}_2$
            if $\max(\|\mathcal{S}_1\|, \|\mathcal{S}_2\|) < \mathcal{L}_{12}^*$ then
                $\mathcal{L}_{12}^* = \max(\|\mathcal{S}_1\|, \|\mathcal{S}_2\|)$
                Store improved solution $\mathcal{V}_1^*, \mathcal{V}_2^*$
            move $\boldsymbol{t}_{i_1}$ back to $\mathcal{V}_1$
    else
        for $i = 1$ to $M_2$ do
            // case $\|\mathcal{S}_2\| > \|\mathcal{S}_1\|$ proceeds analogously
```

Compared to the similar *O-contracts* introduced in [129], the allocation presented here is not individually rational (one agent's cost might increase) but pairwise rational, i.e., we favor the improvement of the pairs min max criterion.

4.3.2 Analysis of the Assignment Procedure

Optimizing the overall objective in (4.4) can be realized by performing the pairwise optimization between all neighbors, either with a predefined deterministic scheme or with a randomized gossip protocol as described in [30]. In both cases, communication steps can be executed in parallel (for teams of $n \geq 4$) as long as the signals do not interfere with each other.

To verify the task allocation process we show that it will always reach a stable state (or equilibrium) in a finite number of steps, first for two robots and then in the general case of n robots.

For the following analysis we focus on a deterministic communication and make some basic assumptions:

- For simplicity we initially assume that all robots are within each others communication range r_c.
- We assume a static scenario during the assignment where the robot and target positions are fixed and no two targets or robots coincide.
- If the assignment procedure is called, the total number of targets will be $0 < M = \sum_{i=1}^{n} M_i < \infty$.

As a consequence, a **state** in the assignment process is uniquely defined by the current partition $\mathcal{P}$, i.e., the targets assigned to each robot.

Definition 4.1. We define one **communication round** as a complete sequence of pairwise optimization steps until each robot has communicated with every other, i.e., one communication round contains $n-1+n-2+\cdots+1=\frac{n^2-n}{2}$ calls of Algorithm 1.

Definition 4.2. An **equilibrium partition** $\mathcal{P}^\epsilon$ of the system is defined by a state where no further improvement can be obtained by application of Algorithm 1 between any two robots. Or in other words, if the state of the assignment is identical before and after one communication round, the system is in a stable state.

We can now formulate the following

Proposition 4.2 (Termination of Algorithm 1)**.** *In a subsystem of two robots i and j, given the assumptions described above and any initial partitioning $\mathcal{P}_{ij}^0$, the application of Algorithm 1 will always lead to an equilibrium partition $\mathcal{P}_{ij}^\epsilon$ and termination of the algorithm in a finite number of iterations. An upper bound for the total number of iterations (including the inner loop) is given by $(M_i + M_j) \cdot 2^{M_i+M_j}$.*

Proof. To exclude the trivial case, we assume that the number of targets $M_i + M_j > 0$. The algorithm will only continue if the cost $\mathcal{L}^*_{ij}$ can be strictly decreased by moving one target. This makes it impossible to reach any particular assignment state twice, since returning to a previous state would require the cost to increase. The total number of possible states is limited as given in (4.6) and the number of cost evaluations performed in each invocation of the inner loop is limited by $\max(M_i, M_j)$. Combining the two statements yields the conclusion above. □

This leads to the main result:

Theorem 4.1 (Convergence of the Overall Assignment)**.** *In a system of* n *robots, given the assumptions above and any initial partitioning* $\mathcal{P}^0$*, the assignment process will always reach an equilibrium partition* $\mathcal{P}^\epsilon$ *in a finite number of communication rounds, which is limited by the number of possible assignments* n^M*.*

Proof. Similar to the previous proof, we show that it is impossible to reach a previous state. For a transition to a new state to occur, one agent's cost (call it i) will decrease while the other agents cost (call it j) will increase or stay the same by applying Algorithm 1. Now there are two possible situations: 1) Agent i was the one with the overall highest cost, making it impossible to reach the previous state because the max of all costs can never increase or 2) agent i was not the one with the overall highest cost. Continuing with case 2), to reach the previous state, agent i would have to get back to a higher cost, which is only possible by communicating with a different agent that has higher cost (call it z). Now either z is the agent with the highest cost, making it impossible to reach the previous state if it moves a target to i, or case 2) is repeated until the agent with the highest cost is reached. At least one state transition will occur by calling Algorithm 1 during one communication round (except for the last communication round where $\mathcal{P}^\epsilon$ is reached), possibly more, so in a worst case scenario, the number of communication rounds is upper bounded by the number of possible assignments. Considering this limitation and incorporating Proposition 4.2 completes the proof. □

Remark 4.2. The convergence result will hold for any order of communication in one communication round and also for cases with more restricted communication, but the assignment may lead to different equilibrium partitions.

In practice the number of communication rounds is much lower than the upper bound. We present empirical results about this in Section 4.5.1.

4.4 Exploration Framework

To evaluate the proposed optimization, we implemented an exploration framework in C++ that allows for comparison with other related approaches. In this section, we describe how the target points are generated and the process of robot motion and individual target selection is explained thereafter.

4.4.1 Frontier Segmentation

Since we consider a polygonal environment instead of a grid map, there is no predefined discretization of the frontier that would provide straightforward target candidates. Therefore we apply the following segmentation: a connected frontier component with length l_f is separated uniformly into line segments of length l_{seg} determined by

$$l_{\text{seg}} = \frac{l_f}{n_{\text{seg}}} \quad \text{with} \quad n_{\text{seg}} = \left\lceil \frac{l_f}{k_f r_e} \right\rceil, \tag{4.7}$$

where n_{seg} denotes the number of segments.

Target points are simply set to the center of each segment. Free parameter $k_f \in \mathbb{R}$ allows adaptation of the segment length and its choice depends on several factors. A low number of longer segments (bigger values of k_f) ensures that target points are not too close to each other and reduces the complexity of the assignment. On the other hand, segments should not be too long to make sure that each frontier component will be explored by visiting its target point. Further, choosing too few segments compared to the number of robots eliminates the variability for optimization. Intuitively, values between $k_f = 1$ and $k_f = 2$ corresponding to $l_{\text{seg}} \leq r_e$ and $l_{\text{seg}} \leq 2r_e$ make sense, as the segments will always be contained in the circle of radius r_e, and have proven to be suitable in our evaluations. An example of a segmentation with $k_f = 1.5$ is shown in Figure 4.4.

4.4.2 Robot Motion and Target Selection

As stated in Section 4.2, the robot motion is modeled by a single integrator, a commonly used approximation for holonomic robots. Investigations with more complex robot models are open for future work. A robot is allowed to move up to 0.1 m at constant speed in between (re-)assignment steps and the amount of assignment steps is counted as iterations. The

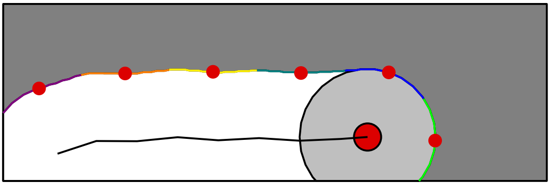


Figure 4.4: Frontier segments in different colors, segmented according to (4.7)

overall number of iterations until the exploration is complete is therefore proportional to the expected exploration time.

For path planning and obstacle avoidance we use the well-known visibility graph method combined with a contraction of the environment [84]. This approach guarantees optimal shortest path calculations and a safety distance to all obstacles. Since it is a combinatorial method (as characterized in [77]), the computational complexity increases quickly for more complex environments with many obstacles. But this is also true for most other methods and we did not have any issues with computation times in the selected scenarios.

If a robot has more than one target assigned without a specific ordering, a target selection procedure for the individual robots is required to determine which target will be approached first. For this purpose we use the same TSP solver that has been demonstrated for use in exploration in [73]. It provides shorter and more consistent exploration trajectories than a greedy approach as explained in Section 4.1.

4.4.3 Distributed Implementation

For a distributed implementation, a common set of target points is required between the two robots that are communicating. This can be realized by either sharing the currently explored area, so that each robot can determine the target points on a common frontier, or calculating the targets only for the individual maps and then exchanging the target information directly. In the second case, the robots have to inform each other about targets that are in the already explored area and delete those (similar to [155]), but only exchanging this point information is usually less expensive than sharing a complete map. The first approach will allow more informed decisions, so the choice depends on the available bandwidth.

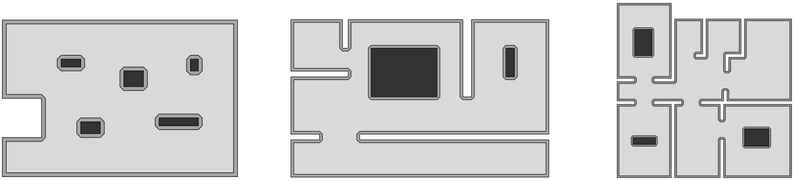
4.5 Results

The evaluation and comparison has been conducted in three different exploration scenarios, representing different types of regions, shown in Figure 4.5. Scenario A is an open space with arbitrarily placed smaller obstacles. Scenario B is a structured space with a long corridor and scenario C is a replica of a building on the basis of a real floor plan. The sizes of the bounding rectangles are $6\,\mathrm{m}\times 4\,\mathrm{m}$, $9\,\mathrm{m}\times 6\,\mathrm{m}$, and $10\,\mathrm{m}\times 10\,\mathrm{m}$ for scenarios A, B, and C, respectively. The exploration time is measured according to the description in Section 4.4.2. Multiple runs for statistical evaluation have been initialized with uniformly random starting locations, but robots start as a group within a distance of $2r_e$ of each other to simulate deployment as a team. We use box plots for the visualization of data with the boxes showing median, lower and upper quartiles and the whiskers depicting minimum and maximum values.

4.5.1 Evaluation of Communication Requirements

An important practical aspect is the actual number of communication rounds required to reach an equilibrium partition as defined in Section 4.3.2. The initial target assignment is determined by shortest distances between the robot positions and targets, while targets in all further iterations are initially assigned based on the distance to a robot and its previously assigned targets. Since this approach provides a good starting solution for subsequent optimizations, a change of assignment is not always necessary.

This is reflected in the simulation results presented in Table 4.1. It shows the average and maximum number of communication rounds required to converge to an equilibrium partition for 50 repetitions of a complete ex-



(a) Scenario A **(b)** Scenario B **(c)** Scenario C

Figure 4.5: Different scenarios for evaluation. Color explanation: □ free space ▨ safety distance ■ obstacle

Table 4.1: Communication rounds and improvement percentage for two different scenarios

	Scenario A			Scenario B		
Number of robots	3	4	5	3	4	5
Avg. rounds	1.28	1.42	1.58	1.24	1.48	1.63
Max. rounds	4	6	5	5	4	5
Improved alloc.	25.2%	36.8%	48.4%	22.8%	44.3%	56.11%

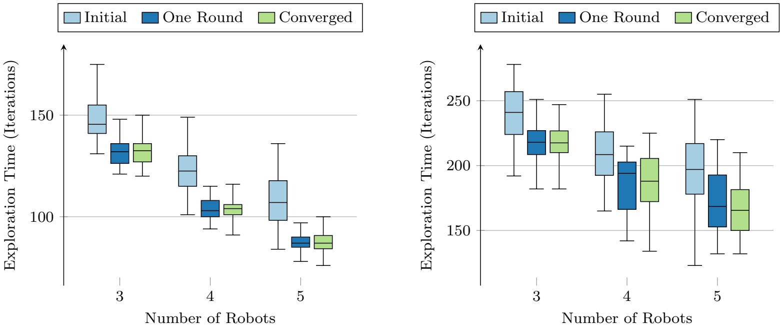
ploration process in two different scenarios. One communication round is always necessary to verify convergence, and in quite a few iterations the current allocation is already an equilibrium partition. The last row of Table 4.1 quantifies this statement, showing the percentage of assignment steps where an improvement to the current allocation was made. For example, in scenario A with 3 robots, in 25.2% of all assignment steps, an improvement to the allocation (which is always based on the target allocation in the previous step, as described above) was found by using the presented algorithm. With an increasing number of robots, this percentage increases significantly, which is reasonable since the complexity and therefore the possibilities to optimize the target distribution also increase.

The results suggest the conclusion that one communication round will be sufficient in most cases and this is in fact confirmed by the results in Figure 4.6. The figure shows exploration times for scenarios A and B with increasing amount of communication, more precisely: only using the initial allocation, one round of communication or until converging to the equilibrium partition. While not using any additional communication produces significantly worse results, using one communication round is always close to the case of full convergence.

Since computation time for additional communication rounds is not an issue, we use the communication until convergence for all further considerations. But if the bandwidth is limited in a real system, the variant with one communication round poses a promising alternative.

4.5.2 Comparison with Related Work

To investigate the performance of the algorithm we implemented several approaches from the literature for comparison:



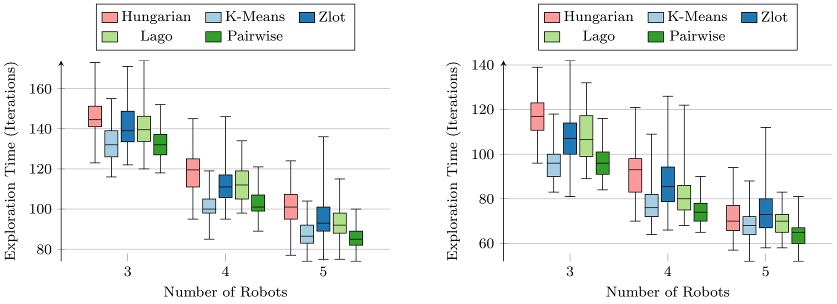
(a) Scenario A **(b)** Scenario B

Figure 4.6: Exploration times with increasing amount of communication

- **Hungarian:** The Hungarian method assigns one target to each robot and has been used for exploration in [149].
- **K-Means:** Based on [33], targets are assigned by a centralized k-means clustering algorithm. The clusters are initialized with the robots as cluster centers, i.e., there is one cluster for each robot and the final clusters determine the points that will be associated with the respective robot. A similar idea has been proposed earlier in [138], where the authors propose a clustering of the unknown space.
- **Zlot:** A market-based approach using marginal cost contracting with self-interested agents [155].
- **Lago:** Auction-based approach from a multi-robot routing context with performance guarantees [74].
- **Pairwise:** Our proposed pairwise optimization strategy from Section 4.3.

On top of the strategies described above, we use a method to assign target locations to idle robots from [33] (if there are any after the original assignment). It is centralized in its original form, but can be modified to be applicable in distributed settings.

Since two of the listed approaches require centralized coordination, i.e., a global knowledge about all target locations, we set r_c of the robots to a value that enables communication at all times to maintain a fair comparison.



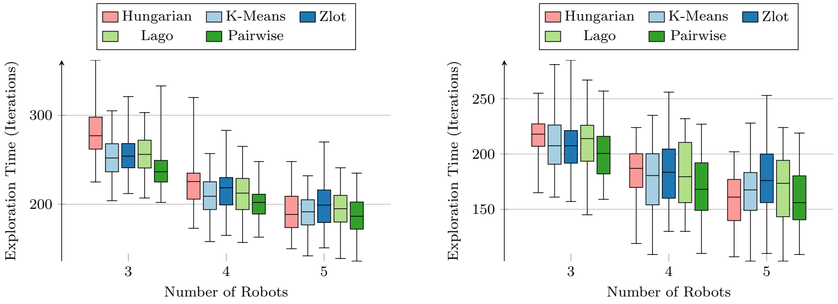
(a) $r_e = 0.5\,\mathrm{m}$ **(b)** $r_e = 0.7\,\mathrm{m}$

Figure 4.7: Exploration times compared for scenario A

The results of the comparison for all three scenarios, with two different sensing ranges in each case, can be found in Figures 4.7 to 4.9. We simulated 100 runs with different starting positions for each combination (method, scenario, sensing range, number of robots), summing up to a total of 9000 runs for this evaluation. Especially in scenario A the K-Means and Pairwise strategies show consistently better results than the other three. This tendency is also present in the other two scenarios, even though not as significant. Noteworthy, the Hungarian strategy performs increasingly better with a higher number of robots and even outperforms K-Means in scenario B for $n = 5$. In comparison with the other two approaches that allow for a distributed implementation (Zlot and Lago), the median of our Pairwise method is always lower. In between Zlot and Lago, the latter has a better overall consistency, especially for higher numbers of robots, where using Zlot can lead to much higher worst case exploration times.

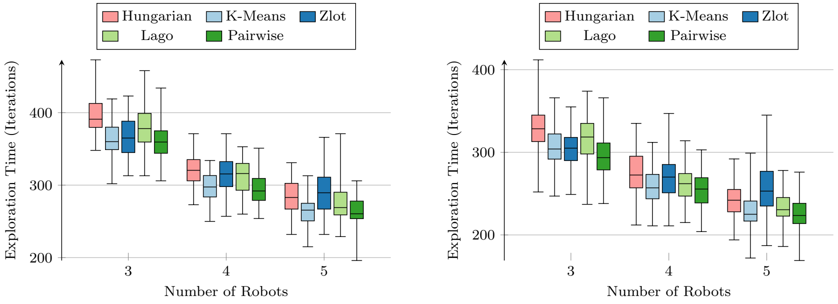
In Zlot's market-based approach, robots that already have a long tour have a higher chance of receiving more targets, because estimated costs are only calculated for insertion or removal of a target to the current tour. The overall length of the tour is not considered at all, which can lead to heavily uneven distribution of target points. This type of allocation tends to minimize the sum of travel distances, but leads to inefficient use of multiple robots and higher overall exploration times.

Computation times of the assignment algorithms have not been an issue for any of the methods used and the time difference between the strategies was not significant enough to motivate further investigation. As already mentioned earlier, the shortest path calculations become increasingly ex-



(a) $r_e = 0.6\,\mathrm{m}$ **(b)** $r_e = 0.8\,\mathrm{m}$

Figure 4.8: Exploration times compared for scenario B



(a) $r_e = 0.8\,\mathrm{m}$ **(b)** $r_e = 1.0\,\mathrm{m}$

Figure 4.9: Exploration times compared for scenario C

pensive for complex or bigger maps, but limiting the number of target points to reduce effort was not necessary in the tested scenarios. To give an example, the average time for an assignment update including path calculations during a typical run of scenario C was 297 ms with a peak value of 712 ms on a 2.5 GHz CPU. Parallelization has not been used, i.e., all computations for the individual robots were executed sequentially.

4.6 Discussion

With the objective to improve coordination in multi-robot exploration, the minimum spanning tree length is used as a cost criterion and optimized in a pairwise manner. The implemented algorithm facilitates a balanced partitioning of workload in highly dynamic scenarios and the simulation results show that this leads to favorable exploration times. The algorithm runs in hundreds of milliseconds on a standard PC, allowing a quasi-continuous replanning, i.e., quick adaptation to changes. Further, the proposed algorithm has lower requirements on communication than similar investigated approaches, promising robustness and scalability.

These aspects offer a solid foundation for application in Stage 2 of the complex search and rescue mission. The robots or exploration units, e.g., teams of UAVs and UGVs, are now able to gather information in unknown environments in a coordinated way and do so efficiently as shown by the evaluations.

5 Coverage Control in Nonconvex Environments

Once the environment is explored, a map is created and possible regions of higher interest or priority have been discovered, the next step for the multi-robot team is to move to favorable locations and monitor the environment for new events that might occur. This task, also referred to as optimal sensor deployment or **coverage control**, corresponds to Stage 3 in the complex search and rescue mission.

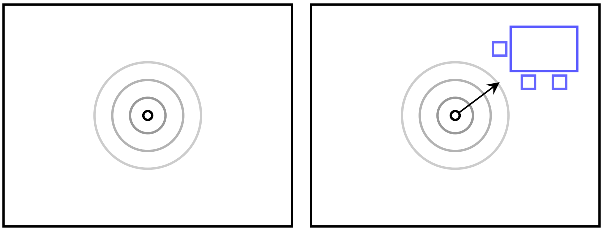
This chapter is structured as follows: The basic idea of coverage control is introduced and illustrated by an example. In the following literature review in Section 5.1.1, problems and drawbacks of current approaches are identified leading to the contribution given in later sections. The problem statement builds on a mathematical description of the underlying concepts and the coverage control algorithm for convex environments is formulated in Section 5.2. Subsequently, a new way to extend the original approach to nonconvex environments is presented and analyzed, including a formulation for systems with limited sensing range, which is important for the applications targeted by this thesis. To verify the solution, simulation results are given for several scenarios. The properties of the presented approach are assessed and compared with close literature, followed by a concluding discussion in Section 5.5.1)

5.1 Introduction

Imagine a situation where multiple mobile robots or sensors - the type of sensor may vary, depending on the task - are placed at possibly random locations in an area of interest, e.g., at the end of an exploration as in the previous chapter. Now the goal is to move the sensors such that they cover the area in an optimal manner. The optimality can be characterized by an objective function that is presented in Section 5.2.3.

---

1) The main results presented in this chapter are also published in Klodt et al. [65].



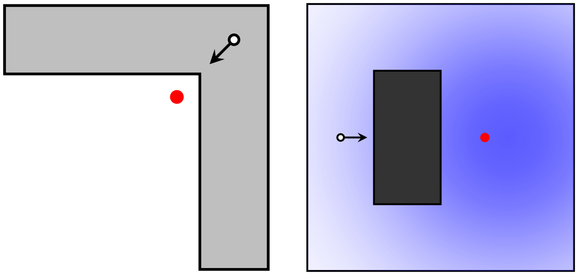
(a) Sensor placement without additional information **(b)** Shift towards more interesting regions

Figure 5.1: Simple example of a coverage task

Figure 5.1 shows a simple example with a single sensor to illustrate the basic idea: Given a limited region with a fixed boundary, an omnidirectional sensor shall be positioned. The sensor has degrading sensor performance with increasing distance, e.g., a microphone. One would intuitively place the sensor in the center of the region (Figure 5.1a). There might be some a priori knowledge about the region that makes certain parts more interesting than others, e.g., a table with chairs. Given this additional knowledge, the sensor placement could be adjusted such that the region of higher interest is covered more thoroughly (as in Figure 5.1b). This basic idea can be extended to multiple sensors, where each sensor has its own share of an environment that should be covered. But first, the important literature specific to the field is discussed.

5.1.1 Literature Review

A popular solution to this problem was presented in [24] and in addition to coverage control it is also referred to as Voronoi coverage. The reason will become clear in the next section. The authors formulate the deployment task as a geometric optimization problem. This kind of resource allocation task was already investigated in the context of information theory and vector quantization [83] and for facility location problems [105] (e.g., where to place radio stations to cover an area as good as possible). The resulting solution is a continuous-time version of the Lloyd algorithm [83], which is introduced in Section 5.2.2. The solution is formulated as a feedback control law that allows distributed, adaptive and asynchronous minimization of the optimization function. A lot of extensions to this approach have been developed including: range limitations [23], utilization



(a) The goal location is outside of the allowed environment

(b) The trajectory would cross through an obstacle on the way to the goal

Figure 5.2: Two cases where the original algorithm fails in nonconvex environments

of anisotropic sensors [78] and estimation of importance of different areas [132]. The coverage framework has also been modified to fulfill different tasks like target tracking [112] and multi-robot exploration [49].

The main drawback of Voronoi coverage is that it only works in convex environments, which is a strong limitation since real world environments are usually nonconvex. The arising complications are already discussed in [9] with several examples and can be summarized in two cases: the sensors goal lies inside the environment but the trajectory crosses through impassable regions, or the goal location lies outside the environment (cf. Figure 5.2). In both cases the robots would collide with the environment and the final configuration is not clearly defined in the second case. These issues have been addressed in several different ways in the literature. The existing approaches are summarized in the following.

In [113], among other extensions, Voronoi coverage is applied to simple polygons without holes using the generalized gradient of the geodesic distance. The authors of [14] propose a transformation for a class of nonconvex environments that allows the application of the coverage algorithm. Mapping back to the original domain gives the solution to the problem. The geometry and a suitable diffeomorphism are assumed to be known beforehand. For a convex set with obstacles the approach converges to the solution of the original problem if it belongs to the interior [13], but in other cases the solution can be suboptimal. Breitenmoser et al. [9] show that for general nonconvex environments a path planning procedure or obstacle avoidance is necessary. They combine the Lloyd algorithm for target point calculation with a local path planner and prove the convergence of the proposed control strategy. The path planner constrains outlying target points to the environment using a projection procedure. A combination of Voronoi coverage and a potential field method is presented in [122]. The approach is a direct extension to the potential field method from [56], adding an additional attractive potential towards the centroid of each robots Voronoi cell. The authors show that their approach yields better final configurations than [56] in selected examples, but do not specify the case where a centroid lies inside an obstacle and provide no theoretical results towards convergence. In [30], Durham et al. propose a coverage strategy on a discretized environment represented by a graph. They define the generalized centroid as the point inside a connected subset that minimizes the sum of weighted distances to all other points. Due to the discretization this point can be found with an exhaustive search or an appropriate heuristic. The authors provide convergence proofs under reduced communication requirements, i.e., only pairwise communication is necessary.

The usage and maximization of visibility sets has been studied extensively in the context of the **art gallery problem**. Solutions to a distributed version of the art gallery problem with mobile sensors have been developed and proven in [39], [102]. The optimization goal is different compared to Voronoi coverage even though it overlaps in some cases.

Also related to our work is the problem presented in [153]. The authors consider a modified coverage framework where overlapping sensor areas increase the quality of coverage and the environment with polygonal obstacles is no longer partitioned into Voronoi cells. The common ground lies in the approach that obstacles attenuate or block the sensing abilities of the robots, leading to a closer analysis of the gradient of the optimization function. The authors show that the use of a generalized gradient is necessary due to a new term that arises from the frontiers to invisible regions.

The combination of Voronoi coverage and limited visibility has only been considered recently by [85] and [90]. In [85] a visibility-based Voronoi diagram is introduced and the authors propose a projection of the centroid to the closest point inside the visible Voronoi cell. A second algorithm considers frontiers to invisible regions, towards which the robots are directed to uncover new areas. The authors provide simulations but no theoretical guarantees that the robots stay inside the environment and converge to a stable configuration. Marier et al. [90] also use the notion of a visible Voronoi diagram similar to [85] with the difference that they allow disconnected cells. The gradient of the optimization function is analyzed leading to similar results as in [153]. A proof of convergence is given for environments without holes. The problem that agents might leave the environment is again handled by a projection procedure.

5.1.2 Contribution

One of the main drawbacks of existing approaches is that they have to rely on separate path planning or projection procedures to ensure collision avoidance with the boundary of the environment and obstacles. Additionally, in some cases, they only work for certain types of environments.

We present a new approach based on [24] that utilizes a virtual deformation of the environment and visibility sets. The resulting coverage strategy has inherent collision avoidance properties without the use of explicit path planning, obstacle avoidance behavior or projections into the allowable area and works in arbitrary nonconvex environments. Additionally, our method has lower requirements on communication and computation capabilities compared to similar solutions. The advantageous properties of Voronoi coverage are retained, i.e., the solution is distributed and the agents can adapt to failing or new sensor nodes or changing density functions. Furthermore, while we assume vehicles with single integrator dynamics, the method can be extended to other system dynamics following [11].

5.2 Problem Formulation

This section begins with an introduction to underlying concepts, i.e., Voronoi partitions and the Lloyd algorithm, continues with the description of the coverage approach for convex environments followed by a formal problem statement.

5.2.1 Voronoi Partitions

According to [47], the Voronoi partition was first observed by G. Lejeune Dirichlet in 1850 [28] and systematically analyzed by Georgy F. Voronoi in 1907 [144]. It is a convenient way to divide a region into cells based on the location of central points.

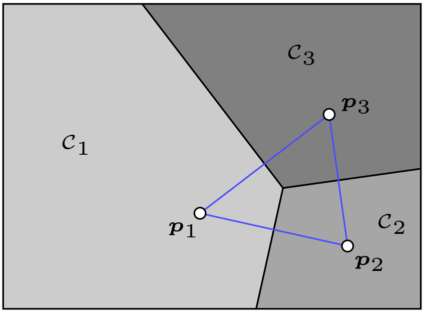


Figure 5.3: Example of a Voronoi partition with three generators and the Delaunay graph in blue

More precisely, a **Voronoi partition** $\mathcal{P} = \{\mathcal{C}_1, ..., \mathcal{C}_n\}$ of environment $\mathcal{Q} \subset \mathbb{R}^2$ can be defined by a set of n **generator points** $\boldsymbol{p}_1, ..., \boldsymbol{p}_n \in \mathcal{Q}$. One Voronoi cell $\mathcal{C}_i$ is the set of all points closer to $\boldsymbol{p}_i$ than to all other generators, i.e., $\mathcal{C}_i = \{\boldsymbol{q} \in \mathcal{Q} \mid \|\boldsymbol{q} - \boldsymbol{p}_i\| \leq \|\boldsymbol{q} - \boldsymbol{p}_j\|, \forall j \neq i\}$. According to this definition, the union of all cells is $\mathcal{Q}$ and the interiors of any two distinct cells share no common points.

A structure dual to the Voronoi partition is the **Delaunay graph**. It is formed by connecting all generator points that have common cell boundaries as shown in Figure 5.3.

In the upcoming paragraphs, it is shown that the Voronoi partition provides an optimal way to separate a given environment into sections for mobile sensors to cover. Furthermore, the Delaunay graph characterizes the communication necessary between the individual agents. This is explained in detail in Section 5.2.3.

5.2.2 Lloyd Algorithm

To understand the movement patterns that are used in Coverage Control it is helpful to look at the **Lloyd algorithm**, a tool for vector quantization originally developed in the context of information theory [83]. Today, it is mostly used and well-known for cluster analysis and often referred to as k-means clustering [87, 139]. It has also been applied to other resource allocation tasks like facility location problems in operational research [105], e.g., where to place radio stations to cover an area as good as possible. In the latter case it can be visualized as a geometric optimization prob-

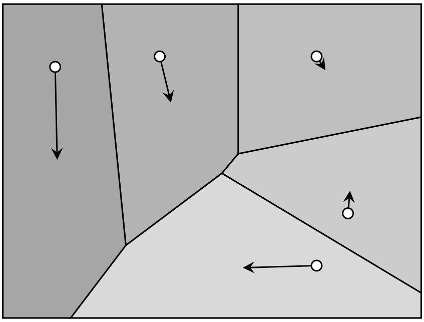


Figure 5.4: A configuration with five sensors and the locations of the centroids where the generator points will be moved to in the next iteration when applying the Lloyd algorithm

lem, which is much closer to the sensor placement task. However, the mathematical foundations stay the same in all cases.

Definition 5.1. (Centroid) A **centroid** or **center of mass** of a region $\mathcal{C} \subset \mathbb{R}^2$ weighted by a density function $\phi : \mathcal{C} \to \mathbb{R}_0^+$ is given by

$$\boldsymbol{c} = \frac{1}{M_c} \int\limits_{\mathcal{C}} \boldsymbol{q}\phi(\boldsymbol{q})d\boldsymbol{q} \quad \text{with} \quad M_c = \int\limits_{\mathcal{C}} \phi(\boldsymbol{q})d\boldsymbol{q}, \tag{5.1}$$

where M_c is the ϕ-weighted area of $\mathcal{C}$.

Note that a weighting of the region is optional and ϕ can assumed to be constant if not further specified.

Using the above definition, the Lloyd algorithm is stated as follows.

- Given n generator points with positions $\boldsymbol{p}_1, ..., \boldsymbol{p}_n \in \mathcal{Q}$, compute the Voronoi partition $\mathcal{P}$ of $\mathcal{Q}$.
- Move the positions of the generators to the centroids of their respective Voronoi cells $\mathcal{C}_i$.
- Repeat these two steps until convergence.

The process is illustrated in Figure 5.4.

In case of discrete data, e.g., for data clustering, the cluster centers have to be initialized as generators at the beginning. Then, the data points are assigned to its closest cluster center and finally the new centroids are computed by averaging all data points within one cluster.

5.2.3 Coverage in Convex Environments

Transferring this idea to a sensor placement task, Cortes, Bullo and Martinez developed a continuous-time version of the Lloyd algorithm where the mobile sensors positions are used as generators for the Voronoi partition [24]. The idea is to minimize the distance to all locations in the environment or in other words that each location in the region of interest is covered by at least one robot. To formulate this as an optimization problem, some preliminary definitions are required.

Assume a convex region $\mathcal{Q} \subset \mathbb{R}^2$, also referred to as an area or environment, that has to be covered. In this environment, a density function can be defined as $\phi : \mathcal{Q} \rightarrow \mathbb{R}_{\geq 0}$ that assigns each location a scalar value. This value is a measure of (a priori) information about the importance of the location or the probability that an event takes place.

The collection of all sensor positions $\boldsymbol{p}_i \in \mathcal{Q}$ is called **configuration** and aggregated in the $2n$-dimensional vector $\boldsymbol{p} = [\boldsymbol{p}_1^\mathsf{T}, \ldots, \boldsymbol{p}_n^\mathsf{T}]^\mathsf{T}$. Note that as in the previous chapter, homogeneous robot teams are assumed and the same considerations apply when it comes to modeling of exploration units and their sensors (cf. Section 4.2.1).

A nondecreasing performance function $f : \mathbb{R}_{\geq 0} \rightarrow \mathbb{R}_{\geq 0}$ is used to characterize sensing performance. The argument is a distance value and the function usually increases, i.e., it penalizes higher distances.

The task of optimal sensor placement can now be formulated as a minimization of the objective

$$H(\boldsymbol{p}) = \int\limits_{\mathcal{Q}} \min_{i \in \{1,\ldots,n\}} f(\|\boldsymbol{q} - \boldsymbol{p}_i\|)\phi(\boldsymbol{q})d\boldsymbol{q}. \tag{5.2}$$

The density function ϕ acts as a weighting that is multiplied to all performance values. As already mentioned above, it can simply be set to a constant if there is no a priori knowledge about the environment.

The minimum operator inside the integral chooses the sensor position $\boldsymbol{p}_i$ that is closest to the current location $\boldsymbol{q}$. Using the definition of the Voronoi partition, (5.2) can be rewritten to

$$H(\boldsymbol{p}) = \sum_{i=1}^{n} \int\limits_{\mathcal{C}_i} f(\|\boldsymbol{q} - \boldsymbol{p}_i\|)\phi(\boldsymbol{q})d\boldsymbol{q}. \tag{5.3}$$

and, as stated in [24], the Voronoi partition is the optimal partition of $\mathcal{Q}$ with respect to the objective. Considering the following statement from

[24] - *...since the Voronoi partition $\mathcal{P}$ depends at least continuously on $\boldsymbol{p}$, $H(\boldsymbol{p})$ is at least continuously differentiable* - provides the smoothness properties for the next steps.

A necessary condition for (5.3) to be at a minimum is, that the partial derivatives with respect to the sensor positions are zero, i.e., $\frac{\partial H(\boldsymbol{p})}{\partial \boldsymbol{p}_i} = \boldsymbol{0}$. As shown in [29], the derivatives only depend on the ith sensor position and the position of it's neighbors in the Delaunay graph to compute the Voronoi cell $\mathcal{C}_i$, yielding

$$\frac{\partial H(\boldsymbol{p})}{\partial \boldsymbol{p}_i} = \int\limits_{\mathcal{C}_i} \frac{\partial}{\partial \boldsymbol{p}_i} f(\|\boldsymbol{q} - \boldsymbol{p}_i\|)\phi(\boldsymbol{q})d\boldsymbol{q} = \boldsymbol{0}. \tag{5.4}$$

As mentioned earlier, the performance function $f(x)$ is used to penalize higher distances, i.e., locations that are farther away result in a higher cost value under the integral. Obviously, different functions will also lead to different optimization results. Possible choices are to simply use the Euclidean or some other application specific distance metric. A suitable approximation for many real sensors that have a signal-to-noise ratio inversely proportional to distance is $f(x) = x^2$ [24].

Equation (5.4) can now be rearranged to give an interesting result:

$$\begin{aligned}
\frac{\partial H(\boldsymbol{p})}{\partial \boldsymbol{p}_i} &= \int\limits_{\mathcal{C}_i} \frac{\partial}{\partial \boldsymbol{p}_i} f(\|\boldsymbol{q} - \boldsymbol{p}_i\|)\phi(\boldsymbol{q})d\boldsymbol{q} && (5.5)\\
&= \int\limits_{\mathcal{C}_i} \frac{\partial}{\partial \boldsymbol{p}_i} \|\boldsymbol{q} - \boldsymbol{p}_i\|^2\phi(\boldsymbol{q})d\boldsymbol{q} && (5.6)\\
&= \int\limits_{\mathcal{C}_i} -2(\boldsymbol{q} - \boldsymbol{p}_i)\phi(\boldsymbol{q})d\boldsymbol{q} && (5.7)\\
&= -2\left(\int\limits_{\mathcal{C}_i} \boldsymbol{q}\phi(\boldsymbol{q})d\boldsymbol{q} - \boldsymbol{p}_i \int\limits_{\mathcal{C}_i} \phi(\boldsymbol{q})d\boldsymbol{q}\right) && (5.8)\\
&= -2M_{c,i}(\boldsymbol{c}_i - \boldsymbol{p}_i) && (5.9)
\end{aligned}$$

The definitions of centroid and weighted area from (5.1) allow for a neat and geometrically interpretable result. It essentially means that the local minimum points for the optimization function (5.3) are those configurations, where the sensor positions coincide with the centroids of their respective Voronoi cells. The resulting partition is called the **centroidal Voronoi configuration**.

A helpful property is that in a convex environment all Voronoi cells are convex, and therefore, the centroid $\boldsymbol{c}_i$ always lies inside the corresponding cell $\mathcal{C}_i$. Assuming first order dynamical behavior

$$\dot{\boldsymbol{p}}_i = \boldsymbol{u}_i, \tag{5.10}$$

a gradient-based control law to minimize (5.3) can be formulated:

$$\boldsymbol{u}_i = k(\boldsymbol{c}_i - \boldsymbol{p}_i) \propto -\frac{\partial H(\boldsymbol{p})}{\partial \boldsymbol{p}_i}, \quad k > 0. \tag{5.11}$$

Proposition 5.1. *(Continuous-time Lloyd descent [24]) For the closed-loop system induced by equation* (5.11) *and the performance function* $f(x) = x^2$, *the sensors location converges asymptotically to the set of critical points of* $H(\boldsymbol{p})$ (5.3), *i.e., the set of centroidal Voronoi configurations on* $\mathcal{Q}$. *Assuming this set is finite, the sensors location converges to a centroidal Voronoi configuration.*

Proof. Differentiating (5.3) with respect to time and inserting (5.11) yields

$$\dot{H}(\boldsymbol{p}(t)) = \frac{dH(\boldsymbol{p})}{dt}\frac{d\boldsymbol{p}}{dt} = \sum_{i=1}^{n} \frac{\partial H}{\partial \boldsymbol{p}_i}\dot{\boldsymbol{p}}_i \tag{5.12}$$

$$= \sum_{i=1}^{n} -2M_{c,i}(\boldsymbol{c}_i - \boldsymbol{p}_i)k(\boldsymbol{c}_i - \boldsymbol{p}_i) \tag{5.13}$$

$$= -2k\sum_{i=1}^{n} M_{c,i}\|\boldsymbol{c}_i - \boldsymbol{p}_i\|^2. \tag{5.14}$$

Since $\dot{H}(\boldsymbol{p}) \leq 0$ for all $\boldsymbol{p} \in \mathcal{Q}$, LaSalle's invariance principle (cf. Theorem 2.2) is applicable and the configuration of sensor locations converges to the largest invariant set $\dot{H}(\boldsymbol{p}) = 0$, which is precisely the set of centroidal Voronoi configurations. □

Note that, depending on $\mathcal{Q}$ and ϕ there are generally multiple centroidal Voronoi configurations and the control law (5.11) does not guarantee convergence to a global minimum.

As suggested in the introduction the problem is now that we want to transfer Voronoi coverage to nonconvex environments.

Problem 5.1. (Nonconvex Coverage) Find a control law based on distributed optimization similar to (5.11) and (5.3) such that all $\boldsymbol{p}_i$ stay inside a nonconvex environment $\mathcal{Q}$ for all t and converge to a stable configuration.

5.3 Extension to Nonconvex Environments

For all further considerations, the environment is defined by a simple polygon $\mathcal{B}$ and the set of simple polygonal obstacles $\mathcal{O}_1, \dots, \mathcal{O}_m \subset \mathcal{B}$ as introduced in Section 2.2.1. The maneuverable domain for the mobile sensors is consequently $\mathcal{Q} = \mathcal{B} \setminus \cup_{i=1}^{m} \mathcal{O}_i$.

The main aspect of our contribution consists of a combination of two modifications to the original problem in (5.3). In conjunction with an approximation of the gradient, the proposed approach leads to the desired behavior. The two modifications are 1) the use of visibility sets and 2) the application of a δ-contraction to the environment.

5.3.1 Visibility

First, let us consider the notion of visibility. For a mathematical introduction, refer to Section 2.2.3. There are several reasons to use visibility sets in this context. The first one is the realistic consideration that sensing capability may be attenuated or completely blocked by obstacles. Using solely the Euclidean distance to characterize sensing performance is no longer viable in the presence of obstacles. Second, the use of visibility allows coverage to be performed in unknown environments, assuming there are no regions with higher priority. Otherwise, knowledge of the density function is still necessary. The third reason is that the limited visibility actually helps to provide a guarantee that the robot stays inside the environment.

Following these thoughts, it makes sense to define a new Voronoi partition tailored to our visibility-limited setting. One option is the usage of the geodesic distance as in [113] and limiting the geodesic Voronoi cells to the visibility set. Another option is the definition proposed in [85] and [90], allowing points closer to one robot to be assigned to another if they are invisible to the former.

Instead, we propose the use of a simpler, visibility-limited Voronoi cell (VLVC)

$$\mathcal{C}_i^{\star} = \{\boldsymbol{q} \in \mathcal{S}^{\star}(\boldsymbol{p}_i) \mid \|\boldsymbol{q} - \boldsymbol{p}_i\| \leq \|\boldsymbol{q} - \boldsymbol{p}_j\|, \forall j \neq i\}. \tag{5.15}$$

This definition describes an intersection between the Voronoi cell and the visibility set of each robot as shown in Figure 5.5. Accordingly, the visibility-limited Voronoi diagram is the set $\mathcal{P}^{\star} = \{\mathcal{C}_1^{\star}, \dots, \mathcal{C}_n^{\star}\}$. Some regions that might be visible are disregarded and treated as invisible, but the computation and communication requirements are reduced compared to

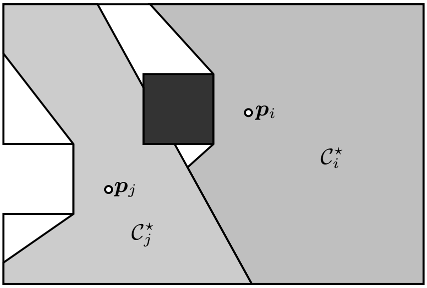


Figure 5.5: Visibility-limited Voronoi cells in a nonconvex environment

other definitions (cf. Section 5.4.2). Range requirements for the communication remain the same as in convex environments, i.e., communication is necessary between neighbors in $\mathcal{P}$.

Note that $\mathcal{P}^\star$ is no longer always a partition of $\mathcal{Q}$ since $\cup_{i=1}^n \mathcal{C}_i^\star \subseteq \mathcal{Q}$. The set of invisible points is $\mathcal{Q}_0 = \mathcal{Q} \setminus \cup_{i=1}^n \mathcal{C}_i^\star$.

5.3.2 Delta-contraction

The second modification, as mentioned above, is the application of a δ-contraction, also known as growing of obstacles in robotics for collision-free path planning [84]. For an environment $\mathcal{Q}$, the δ-contraction is defined as

$$\mathcal{Q}_\delta = \{\boldsymbol{q} \in \mathcal{Q} \mid \inf_{\boldsymbol{q}' \in \partial\mathcal{Q}} \|\boldsymbol{q} - \boldsymbol{q}'\| \geq \delta\}, \tag{5.16}$$

i.e., the set of all points in $\mathcal{Q}$ with a distance to the boundary of $\mathcal{Q}$ greater than or equal to δ (see Figure 5.6).

A remarkable property of the δ-contraction is the following: for arbitrary small $\delta > 0$, the boundary $\partial\mathcal{Q}_\delta$ of the δ-contraction is continuously differentiable in all $\boldsymbol{q} \in \partial\mathcal{Q}_\delta$ except in the convex vertices. All concave vertices grow by δ, yielding a differentiable circular segment.

Depending on the environment and the value of δ, a connected set $\mathcal{Q}$ can have a disconnected $\mathcal{Q}_\delta$. However, considering the previous remark, if $\mathcal{Q}$ is connected, δ can always be selected in a way that $\mathcal{Q}_\delta$ is also connected.

5.3.3 Derivation of the Objective Function

Given a nonconvex environment $\mathcal{Q}$ as defined earlier, we propose to apply Voronoi coverage to the δ-contraction of the environment in a limited

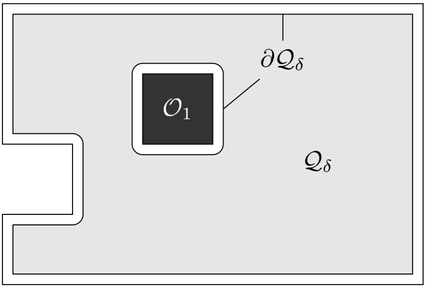


Figure 5.6: Environment $\mathcal{Q}$ and the corresponding δ-contraction

visibility setting. For this purpose, we define the performance function

$$f_{\text{vis}}(\|\boldsymbol{q}-\boldsymbol{p}_i\|) = \begin{cases} \|\boldsymbol{q}-\boldsymbol{p}_i\|^2, & \text{if } \boldsymbol{q} \in \cup_{j=1}^{n}\mathcal{C}_j^\star, \\ D^2, & \text{if } \boldsymbol{q} \in \mathcal{Q}_0, \end{cases} \tag{5.17}$$

with D being a parameter that penalizes invisible areas and controls the uncovering of such regions. Note that f_{vis} is completely defined on $\mathcal{Q}$ since $(\cup_{j=1}^{n}\mathcal{C}_j^\star)\cup\mathcal{Q}_0 = \mathcal{Q}$, with discontinuities at the frontiers to invisible regions that will be specified in the upcoming paragraphs.

Now, applying the δ-contraction to $\mathcal{Q}$, the resulting objective function on the domain $\mathcal{Q}_\delta$ is

$$H_{\text{vis}}(\boldsymbol{p}) = \sum_{i=1}^{n}\int_{\mathcal{C}_{\delta,i}^\star}\|\boldsymbol{q}-\boldsymbol{p}_i\|^2\phi(\boldsymbol{q})d\boldsymbol{q} + \int_{\mathcal{Q}_{\delta,0}} D^2\phi(\boldsymbol{q})d\boldsymbol{q} \tag{5.18}$$

with $\mathcal{C}_{\delta,i}^\star = \{\boldsymbol{q} \in \mathcal{C}_i^\star \mid [\boldsymbol{p}_i,\boldsymbol{q}] \subset \mathcal{Q}_\delta\}$ denoting the VLVC and $\mathcal{Q}_{\delta,0} = \mathcal{Q}_\delta \setminus \cup_{i=1}^{n}\mathcal{C}_{\delta,i}^\star$ denoting the invisible area, both in the new shrunken environment $\mathcal{Q}_\delta$. Unfortunately, the partial derivative of H_{vis} is more complicated compared to (5.4). For a better understanding of the derivations, we consider the coordinates of $\boldsymbol{p}_i = [x_i, y_i]^\mathsf{T}$ separately.

Using an extended form of the Leibniz integral rule for differentiation under the integral sign [34], as the domain of integration is a function of

$\boldsymbol{p}$, gives

$$\frac{\partial H_{\text{vis}}(\boldsymbol{p})}{\partial x_i} = \sum_{j=1}^{n} \int\limits_{\mathcal{C}^\star_{\delta,j}} \frac{\partial}{\partial x_i} \|\boldsymbol{q} - \boldsymbol{p}_j\|^2 \phi(\boldsymbol{q}) d\boldsymbol{q} \tag{5.19a}$$

$$+ \sum_{j=1}^{n} \int\limits_{\partial\mathcal{C}^\star_{\delta,j}} \|\boldsymbol{q} - \boldsymbol{p}_j\|^2 \phi(\boldsymbol{q}) \boldsymbol{v}_j^\top \boldsymbol{n}_j ds \tag{5.19b}$$

$$+ \int\limits_{\mathcal{Q}_{\delta,0}} \frac{\partial}{\partial x_i} D^2 \phi(\boldsymbol{q}) d\boldsymbol{q} \tag{5.19c}$$

$$+ \int\limits_{\partial\mathcal{Q}_{\delta,0}} D^2 \phi(\boldsymbol{q}) \boldsymbol{v}_0^\top \boldsymbol{n}_0 ds, \tag{5.19d}$$

where $\boldsymbol{v}_j = \frac{\partial \boldsymbol{q}(\partial\mathcal{C}^\star_{\delta,j})}{\partial x_i}$ and $\boldsymbol{v}_0 = \frac{\partial \boldsymbol{q}(\partial\mathcal{Q}_{\delta,0})}{\partial x_i}$ are the derivatives of boundary points $\boldsymbol{q} \in \partial\mathcal{C}^\star_{\delta,j}$ and $\boldsymbol{q} \in \partial\mathcal{Q}_{\delta,0}$ with respect to x_i, interpreted as velocities of the moving boundaries. Further, $\boldsymbol{n}_j$ and $\boldsymbol{n}_0$ are the outward facing unit normals on the respective boundaries and ds is the element of arc length. Clearly, all elements of the sum in (5.19a) for $j \neq i$ and the term (5.19c) are zero, leaving only the integral over the i-th VLVC and the boundary terms (5.19b), (5.19d).

Taking a closer look at the boundary of a cell $\mathcal{C}^\star_{\delta,j}$, there are four different parts to distinguish (see also Figure 5.7):

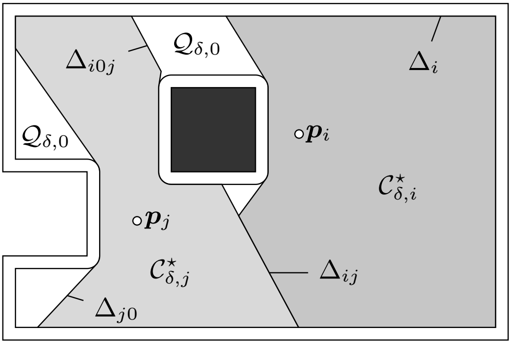


Figure 5.7: Different types of frontiers

1. Boundary points on intersections with $\partial\mathcal{Q}_\delta$, denoted with $\Delta_j = \partial\mathcal{C}^\star_{\delta,j} \cap \partial\mathcal{Q}_\delta$, have a vanishing derivative $\frac{\partial \boldsymbol{q}(\Delta_j)}{\partial x_i} = 0$ for all j in static environments.

2. Boundary points on intersections with neighboring VLVC, denoted with $\Delta_{ij} = \Delta_{ji} = \partial\mathcal{C}^\star_{\delta,i} \cap \partial\mathcal{C}^\star_{\delta,j}$, have nonzero derivatives with respect to x_i in two cases, either $j = i$ or $j \in \mathcal{N}_i$, where $\mathcal{N}_i = \{j \in \{1,\ldots,n\} \mid \Delta_{ij} \neq \emptyset, j \neq i\}$ is the neighborhood of $\boldsymbol{p}_i$. The integrand of (5.19b) is identical in both cases, due to the definition of the VLVCs, with the exception that the outward normals point in opposite directions. This leads to

$$\begin{aligned}&\int\limits_{\partial\mathcal{C}^\star_{\delta,i}\setminus\partial\mathcal{Q}_{\delta,0}} \|\boldsymbol{q}-\boldsymbol{p}_i\|^2\phi(\boldsymbol{q})\boldsymbol{v}_i^\mathsf{T}\boldsymbol{n}_i ds\\ &= -\sum_{j\in\mathcal{N}_i}\int\limits_{\Delta_{ij}} \|\boldsymbol{q}-\boldsymbol{p}_j\|^2\phi(\boldsymbol{q})\boldsymbol{v}_j^\mathsf{T}\boldsymbol{n}_j ds\end{aligned}$$

   and elimination of the corresponding parts of the sum of integrals in (5.19b).

3. The third and remaining part on the boundary of a VLVC is the intersection with the boundary of the invisible area $\partial\mathcal{Q}_{\delta,0}$, denoted with $\Delta_{j0} = \partial\mathcal{C}^\star_{\delta,j} \cap \partial\mathcal{Q}_{\delta,0}$. Those parts result in nonzero terms out of (5.19b) and (5.19d) for $j = i$, and in the case explained in 4).

4. In some cases, the boundary of the Euclidean Voronoi cell $\mathcal{C}_i$ intersects with the boundary between the invisible area and another VLVC, e.g., the part that is labeled with $\Delta_{i0j} = \partial\mathcal{C}_i \cap \partial\mathcal{Q}_{\delta,0} \cap \mathcal{C}^\star_{\delta,j}$ in Figure 5.7. This boundary will move with the position of $\boldsymbol{p}_i$, even though it is not part of the VLVC of robot i.

Concluding the above considerations, the remainder of (5.19) is reduced to

$$
\begin{aligned}
\frac{\partial H_{\text{vis}}(\boldsymbol{p})}{\partial x_i} &= \int\limits_{\mathcal{C}^{\star}_{\delta,i}} \frac{\partial}{\partial x_i} \|\boldsymbol{q}-\boldsymbol{p}_i\|^2 \phi(\boldsymbol{q}) d\boldsymbol{q} && \text{(5.20a)}\\
&+ \int\limits_{\Delta_{i0}} (\|\boldsymbol{q}-\boldsymbol{p}_i\|^2 - D^2)\phi(\boldsymbol{q})\boldsymbol{v}_i^{\mathsf{T}}\boldsymbol{n}_i ds && \text{(5.20b)}\\
&+ \sum_{j\in\mathcal{M}_i} \int\limits_{\Delta_{i0j}} (\|\boldsymbol{q}-\boldsymbol{p}_j\|^2 - D^2)\phi(\boldsymbol{q})\boldsymbol{v}_j^{\mathsf{T}}\boldsymbol{n}_j ds && \text{(5.20c)}
\end{aligned}
$$

with $\mathcal{M}_i = \{j \in \{1,\dots,n\} \mid \Delta_{i0j} \neq \emptyset, j \neq i\}$. Similar considerations hold true for $\frac{\partial H_{\text{vis}}(\boldsymbol{p})}{\partial y_i}$ and are omitted.

Without the moving boundaries to invisible areas Δ_{i0} and Δ_{i0j}, the result in (5.20) would be equal to (5.4), i.e., the well-known move to centroid behavior. The additional terms can be interpreted as movement vectors that lead to revelation of invisible areas, depending on the choice of D. Higher values of D result in a higher weighting of the additional terms compared to the movement towards the center of mass of the VLVC. Low values of D or disregarding the invisible area, i.e., $D = 0$, would actually lead to robots actively reducing visible area to minimize H_{vis}. This can be explained with the sign of the integrand changing dependent on $\|\boldsymbol{q}-\boldsymbol{p}_i\|^2 > D^2$ or $\|\boldsymbol{q}-\boldsymbol{p}_i\|^2 < D^2$, turning the direction of the normal vectors $\boldsymbol{n}_i$.

Unfortunately, (5.20b) and (5.20c) require a high effort to compute, lead to unwanted movements towards the boundary of $\mathcal{Q}$ in several cases and have negligible effect in many other situations. Therefore we propose the following

Assumption 5.1. (Approximation of the gradient) The integral terms arising from the boundaries to invisible regions in $\frac{\partial H_{\text{vis}}(\boldsymbol{p})}{\partial \boldsymbol{p}_i}$ have negligible effect on the desired coverage behavior and the gradient can be approximated by

$$
\frac{\partial H_{\text{vis}}(\boldsymbol{p})}{\partial \boldsymbol{p}_i} \approx \partial H_{\text{approx},i} = \int\limits_{\mathcal{C}^{\star}_{\delta,i}} \frac{\partial}{\partial \boldsymbol{p}_i} \|\boldsymbol{q}-\boldsymbol{p}_i\|^2 \phi(\boldsymbol{q}) d\boldsymbol{q}. \tag{5.21}
$$

Depending on the situation this assumption can be quite strong or very reasonable. We deliberately accept the fact that there will be no active movement towards (uncovering of) invisible areas. Further effects are discussed in the later sections.

Using (5.21), the resulting control law of our approximated coverage strategy is

$$\dot{\boldsymbol{p}}_i = \boldsymbol{u}_i = k(\boldsymbol{c}_i^\star - \boldsymbol{p}_i) \tag{5.22}$$

with the centroid of the VLVC

$$\boldsymbol{c}_i^\star = \frac{\int_{\mathcal{C}_{\delta,i}^\star} \boldsymbol{q}\phi(\boldsymbol{q})d\boldsymbol{q}}{\int_{\mathcal{C}_{\delta,i}^\star} \phi(\boldsymbol{q})d\boldsymbol{q}}. \tag{5.23}$$

$H_{\text{vis}}(\boldsymbol{p})$ is no longer minimized exactly but approximately, meaning that $\frac{\partial H_{\text{vis}}(\boldsymbol{p})}{\partial \boldsymbol{p}_i} \approx 0$ in equilibrium points where $\boldsymbol{p}_i = \boldsymbol{c}_i^\star$.

In contrast to the results in [153] and [90], our approximated gradient $\partial H_{\text{approx},i}$ exists for all $\boldsymbol{p}_i \in \mathcal{Q}_\delta$. Hence, we do not have to rely on the computation of a generalized gradient, as was done in the related work.

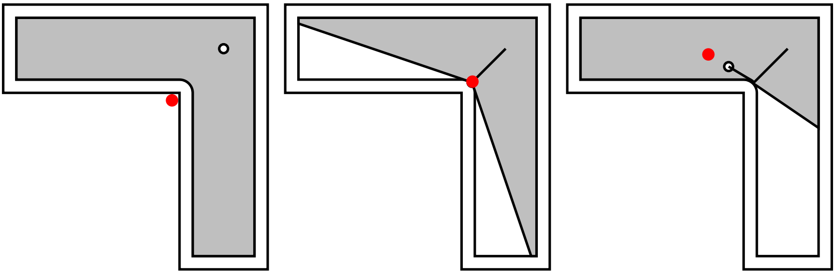
5.3.4 Trajectories

In this subsection, we provide a statement about the trajectories of the proposed system and some comments on convergence.

Theorem 5.1 (Trajectories in nonconvex environments)**.** *For a nonconvex polygonal environment $\mathcal{Q}$ and an arbitrary small $\delta > 0$, continuous application of* (5.22) *results in trajectories of the configuration $\boldsymbol{p}$ that never leave the invariant set $\mathcal{Q}_\delta$.*

Proof. First, pointing out the fact that in convex locations the centroid of $\mathcal{C}_{\delta,i}^\star$ always lies in $\text{int}(\mathcal{C}_{\delta,i}^\star)$ yields the conclusion that collisions with $\mathcal{Q}_\delta$ can only occur in concave locations. Second, applying the δ-contraction to $\mathcal{Q}$ turns all concave vertices in $\partial\mathcal{Q}$ into continuously differentiable concave circular segments in $\partial\mathcal{Q}_\delta$. Approaching concave locations $\boldsymbol{q}$ on the boundary $\partial\mathcal{Q}_\delta$, the boundary of the visibility-limited Voronoi cell $\mathcal{C}_{\delta,i}^\star$ facing $\boldsymbol{q}$ approaches the tangent in $\boldsymbol{q}$ and the centroid inevitably approaches $\text{int}(\mathcal{C}_{\delta,i}^\star) \subset \mathcal{Q}_\delta$. Hence, no trajectory exists that leaves $\mathcal{Q}_\delta$. □

The process of approaching a concave location is illustrated in Figure 5.8 in a scene with constant density. The circle symbolizes the robot position and the red dot indicates the centroid of the VLVC. After reaching an unstable equilibrium in Figure 5.8b, i.e., a point where the gradient is theoretically zero, the robot moves to a stable location on either side of



(a) The initial configuration **(b)** An unstable equilibrium **(c)** Moving to final location

Figure 5.8: Approaching a concave location: the robot positions is shown as a black circle and the red dot indicates the centroid of the VLVC.

the symmetrical scene, due to numerical inaccuracies. In this context, another advantage of the δ-contraction becomes clear: it offers guaranteed collision avoidance with the environment by setting δ to the physical radius of the robots.

Even though convergence to a stable centroidal configuration seems obvious, since all robots move towards the centroids of their respective cells, a formal proof is still missing.

5.3.5 Limited Sensing Range

In a realistic setting, alongside the attenuation of sensing abilities through obstacles, sensors usually have a limited sensing range r, i.e., measurements beyond r are not accurate enough or yield no results at all. This kind of approach is related to the *mixed distortion-area problem* in [11], now combined with our modifications from Sections 5.3.1 and 5.3.2. The objective function for range limited coverage on visibility sets is

$$H_{\mathrm{vis},r}(\boldsymbol{p}) = \sum_{i=1}^{n} \int_{\mathcal{C}^{\star r}_{\delta,i}} \|\boldsymbol{q} - \boldsymbol{p}_i\|^2 \phi(\boldsymbol{q}) d\boldsymbol{q} + \int_{\mathcal{Q}^{r}_{\delta,0}} r^2 \phi(\boldsymbol{q}) d\boldsymbol{q}, \tag{5.24}$$

with the r-limited visibility sets $\mathcal{C}^{\star r}_{\delta,i} = \{\boldsymbol{q} \in \mathcal{C}^{\star}_{\delta,i} \mid \|\boldsymbol{q} - \boldsymbol{p}_i\|^2 \leq r\}$ and the uncovered area $\mathcal{Q}^{r}_{\delta,0} = \mathcal{Q}_\delta \setminus \cup_{i=1}^{n} \mathcal{C}^{\star r}_{\delta,i}$. This definition of $H_{\mathrm{vis},r}$ has the convenient property that all additional boundary terms that lie on distance r to a robot position are canceled out in the gradient $\frac{\partial H_{\mathrm{vis},r}(\boldsymbol{p})}{\partial \boldsymbol{p}_i}$.

Hence, the remaining gradient is similar to (5.20) and we can use the same assumption as in Section 5.3.3 to obtain the control law

$$\dot{\boldsymbol{p}}_i = \boldsymbol{u}_i = k(\boldsymbol{c}_i^{\star r} - \boldsymbol{p}_i). \tag{5.25}$$

Similar to the unlimited case, (5.25) moves the robots to the centroids of their respective visibility sets, herein limited by the radius r. Theorem 5.1 from Section 5.3.4 can be applied analogously.

An additional advantage of this range limited setting is a reduced requirement on the communication range. If robots have distance $\|\boldsymbol{p}_i - \boldsymbol{p}_j\| > 2r$, the visibility sets do not intersect and there is no reason to communicate. In a worst-case scenario with unlimited sensing range, the communication range necessary is the diameter of the complete environment.

5.4 Results

5.4.1 Simulations

In this section, we demonstrate the effectiveness of our approach in a variety of simulated environments. The δ-contraction has been applied to all scenarios with a value of $\delta = 0.15\,\mathrm{m}$. Figures of scene A, B, and C always illustrate the final, converged sensor positions and the trajectories leading to these positions. As the first example, Figure 5.9 and 5.10 show the comparison of a similar convex and nonconvex environment of size $6\,\mathrm{m}\times 4\,\mathrm{m}$ with the same starting configuration of 5 sensors in the top left corner, both cases with a uniform density $\phi(\boldsymbol{q}) = 1$. In addition, Figure 5.11 shows the progression of H_{vis} corresponding to the trajectories in Figure 5.10. Apparently, H_{vis} is minimized despite our approximation.

Figure 5.12 shows a coverage scenario with several obstacles in a rectangular domain of $15\,\mathrm{m}\times 10\,\mathrm{m}$. The same environment, modified with a Gaussian density function shown in blue, is covered by agents with a limited sensing range of $r = 3\,\mathrm{m}$ in Figure 5.13. The Gaussian function has a variance of $\sigma_x^2 = \sigma_y^2 = 4$ in both x- and y-direction with peak height 0.5 located in the bottom right part of the environment.

A corridor-like region with a constant density is considered in Figure 5.14, where sensors start in the far left part of the domain and spread out all the way up to the right. The progression of H_{vis} in Figure 5.15 shows that our approximated gradient $\partial H_{\mathrm{approx},i}$ does not strictly minimize the original objective. Nevertheless a good final configuration can be achieved.

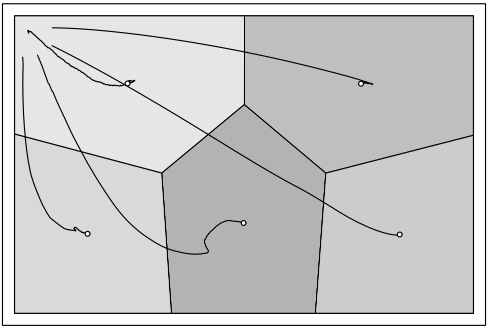


Figure 5.9: Convex scene A with five sensors

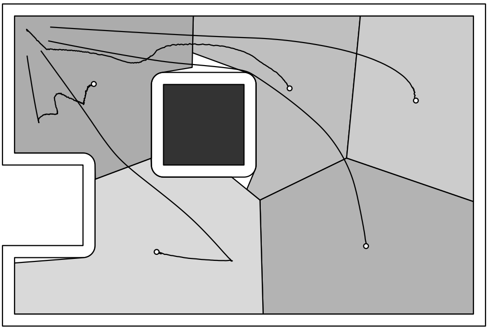


Figure 5.10: Nonconvex scene A with five sensors

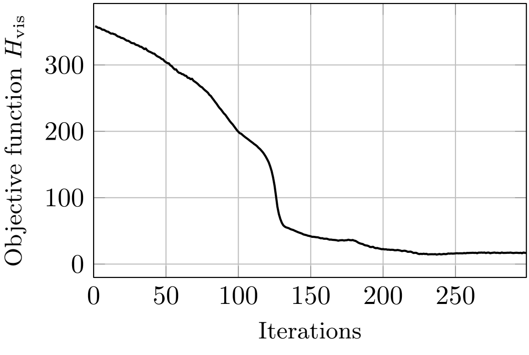


Figure 5.11: Progression of H_{vis} corresponding to Figure 5.10

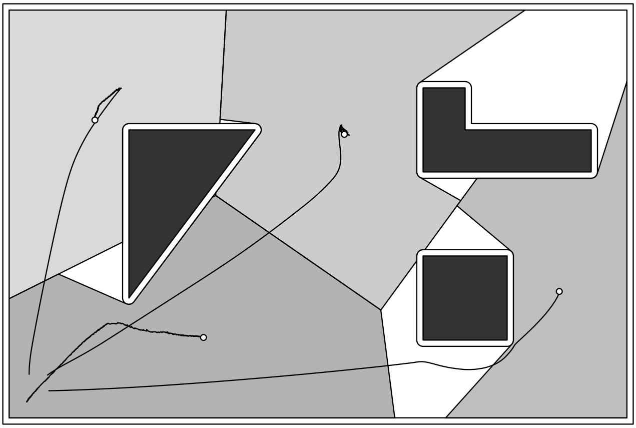


Figure 5.12: Nonconvex scene B with four sensors

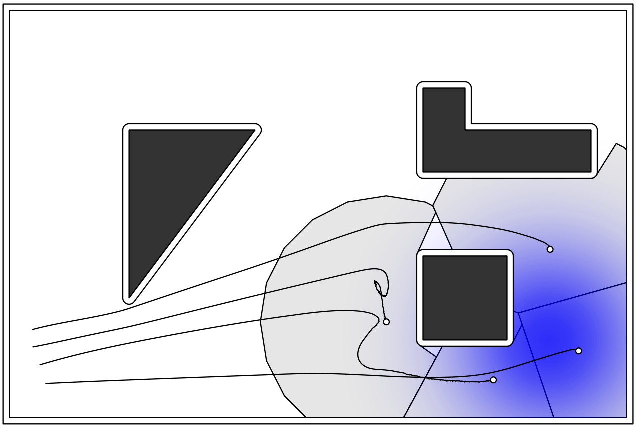


Figure 5.13: Nonconvex scene B with limited sensing range and a nonuniform density

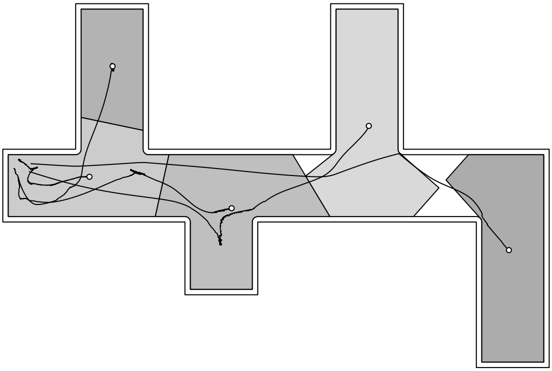


Figure 5.14: Nonconvex scene C with five sensors

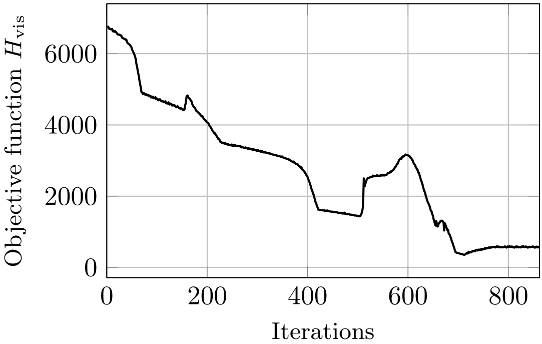


Figure 5.15: Progression of H_{vis} corresponding to Figure 5.14

In all our simulations, there was no situation where positions did not converge to a stable centroidal configuration.

5.4.2 Comparison

As a final assessment, the properties of our nonconvex Voronoi coverage are discussed and compared with regard to optimality, computational effort and communication load. Computational effort and communication load are important factors in a distributed environment, as evaluation is done on the individual sensors, where both computation as well as communication capabilities are limited.

One important point is the choice of partition. The computation of the VLVC proposed in Section 5.3.1 is a simple intersection operation, whereas other authors have to rely on a discretization of the environment combined with a flooding method (e.g., [85]) for cell computation. Marier et al. [90] use the same partition as presented in [85]. Another additional effort is the treatment of disconnected cells, which do not arise in our case. The partition also has an influence on the required communication: Using the VLVCs, the sensors only have to communicate their position to Voronoi neighbors. Propagation of the visibility boundaries is not necessary in our case, as opposed to the compared approaches. The gain in optimality choosing the more complex partition remains uncertain. Using the simplified notion of the visibility-limited Voronoi diagram compared to [85] and [90] does not mean agents do not see the area or cannot sense there, but we only disregard it in the computation of the control law.

The computation of the control input (5.22) is as simple as in the original approach from [24] in known environments. In unknown environments, the δ-contraction has to be computed online, but this case is rarely considered in coverage literature. In [85], an additional path planning and projection procedure is necessary at all times. Calculating the control from [90] is even more costly, due to integrals over the moving visibility frontiers and use of a generalized gradient. Further, the parameter D has to be chosen and adapted appropriately to the size of the environment.

Regarding optimality, reconsider the example in Figure 5.8. Optimizing H_{vis} exactly would always lead to convergence to a minimum close to the position shown in Figure 5.8b (provided an appropriate choice of D), with trajectories possibly leaving $\mathcal{Q}_\delta$ temporarily. Using $\partial H_{\text{approx},i}$ leads to an unstable equilibrium and two stable local minima, one of which is shown in Figure 5.8c. This is a case where the approximated solution can be relatively far away from the exact, in many other situations the

Table 5.1: Comparison of visibility-based Voronoi coverage. The symbols characterize equal (∘), inferior (−), or superior (+) performance in the respective category.

	Lu et al. [85]	Marier et al. [90]	This approach
Optimality	∘	+	∘
Computat. effort	∘	−	+
Communic. load	−	−	+

exact and approximated minimum locations are closer together. Generally speaking, in scenes with high nonconvexity and a low number of robots, the optimality is reduced. On the other hand, especially in the case of limited sensing range, frontiers to invisible regions have a low impact and the approximated solution is advantageous. Furthermore, the optimization itself is nonconvex in all nontrivial cases, even for convex environments, i.e., the configuration converges only to a local minimum with a gradient method. Still, the exact method from [90] has the advantage with respect to optimality. The results of the discussion are summarized in Table 5.1, showing a comparison to the two closely related approaches.

5.5 Discussion

The positioning of mobile sensing units to optimal locations is the third and final step in the initially presented search and rescue scenario. The sensors are now able to observe and respond to any further events that may occur. To this end, Voronoi coverage has been transferred to nonconvex environments successfully. The proposed approximate solution has advantageous properties with regard to communication and computation requirements and obstacle avoidance behavior as shown in the comparison above.

6 Summary and Outlook

In this final chapter, the results established in this thesis are reviewed and evaluated and considerations on further developments are presented.

6.1 Overview

Thinking back to the **introduction**, we mused about the general motivation to develop robotic or automatic systems and found inspiration dating back to ancient Greek mythology and philosophy. With an idealistic vision about abolition of serf labor and liberation of mankind, we moved to more concrete aspects and applications, where robots are being used in today's world.

Narrowing the focus further, an extensive **literature** review on multi-agent systems and cooperative robotics was presented, providing insights into the current state of the art. With this background about prevailing research areas, the specific goals and contributions of the dissertation were outlined in Section 1.3. A complex search and rescue mission consisting of three stages was given as a motivating example, incorporating motion coordination, information gathering and monitoring. These stages, corresponding to chapters 3, 4 and 5, are shown in Figure 6.1 as a reminder.

Beforehand, the mathematical **foundations** in graph theory, computational geometry and dynamical systems were introduced as a basis to understand the subsequent chapters. Important topics including spanning trees, sets and visibility and stability criteria have been covered as well.

The common aspects that formed the **central point** in this thesis are the coordination and cooperation between multiple robots. Starting with

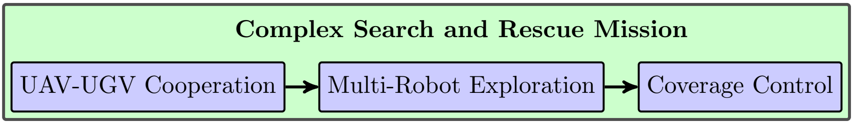


Figure 6.1: The three stages of the motivational example mission

two robots in the simplest case, the ideas can be extended to an arbitrary amount of mobile units in the scalable, distributed settings of exploration and coverage.

Other **challenges** that had to be considered even if they are not part of the main research questions include sensing, path planning, communication and obstacle avoidance. While sensing and path planning were more prevalent in the first stage, communication and obstacle avoidance played a more important role in the latter two stages.

The main method to learn more about scenarios presented in this thesis are simulations. They are an extremely useful tool for several reasons: they allow to investigate systems in a much larger scale, number of repetitions and variety of situations then would ever be possible with real robots, and in a much shorter amount of time. Disadvantages are the simplifications and assumptions that have to be made, i.e., conclusions may not be directly transferred to a real system.

6.2 UAV-UGV Cooperation

Although many different cooperative UAV-UGV applications and scenarios have been investigated, the literature research in Chapter 3 revealed an interesting fact. None of the existing approaches fully uses the region of visibility in which the UAV can move while still being able to detect the UGV. Applying this idea, two approaches have been developed and verified. More specifically, the problem of finding a stabilizing control law for the UAV that minimizes the coverage error function while maintaining visibility to the UGV was solved.

One of the first simplifications necessary to explore this idea was choosing a suitable model for the vehicles. Even though the double integrator is a suitable model for quadcopters during low velocity and acceleration it does cause deviations in the simulations that are hard to estimate. One of the next steps for future developments would be to simulate the presented algorithms with a more accurate quadrotor model (e.g., [128]) and evaluate the results compared to a simpler UAV model.

As a basis for the UAV movement, a dynamic coverage strategy from the multi-agent literature has been adopted and extended in two aspects: a density function that represents the coverable region around the UGV has been introduced and the original approach has been extended to double integrator systems. The extensions have first been developed and proven theoretically and then verified in simulation. To fulfill the visibility part of the problem statement, a tracking controller, employing standard techniques from control theory, has been introduced and adapted to the UAV-UGV setting. The biggest challenge – combining the two control parts in a useful way – has been approached with weighting functions to balance out the control inputs. A threshold value has been established that guarantees compliance with the visibility constraint. Different types of weighting functions have been evaluated to find the most useful one for the posed application. Finally, simulations and comparison with a virtual point tracking method as a more simple baseline approach are presented. The results show that the more sophisticated method – combined dynamic coverage and tracking – also promises better results with regards to coverage error minimization.

For future work, more simulation experiments would also be interesting with a specific higher level task for the UGV. In the simplest case, one could add obstacles on the ground and the UGV has to find the shortest path from a starting position to a goal location. To evaluate this experiment, the UGV with and without additional information from a UAV could be simulated to see the effects. More advanced higher level tasks like exploring an unknown environment with obstacles or adding air obstacles are possible for further steps towards more realistic scenarios.

Another idea is the addition of more UAVs that assist the UGV at the same time. The area around the UGV could be covered faster and with more reliability, if the challenges of fusing and communicating the data can be solved adequately. Variations of this idea are, that one UAV follows the other – again within a visibility range – or both UAVs follow the UGV.

To increase robustness, backup strategies in case of failure of air or ground vehicle would pose a promising research task as well.

6.3 Multi-Robot Exploration

After investigating the cooperation between two individual, heterogeneous robots, the next stage of the mission was concerned with coordinating an arbitrary number of robots for exploration. Starting from the prevalent frontier-based approach and an extensive literature research, an idea towards more dynamic and even distribution of goal locations has been introduced. Based on the minimum spanning tree as an estimator for the individual robots path lengths, a pairwise optimization procedure was proposed, that simplifies the highly complex assignment task to an iterative assignment between couples of robots. The contributions include proofs for the pairwise and overall assignment procedure as well as extensive simulations to evaluate and compare with related approaches.

Drawbacks of these comparative evaluations are that simulations can only indicate the performance of a real system but do not allow for definite statements. Additionally, parameter tuning always plays a role and can change the outcome of simulation experiments significantly. Further steps towards more conclusive results would be to conduct more realistic simulations (e.g., with more detailed robot and environment models, probabilities, realistic physics, different types of terrain etc.) and eventually experiments with hardware. The influence of limited communication on the performance of the proposed pairwise coordination algorithm is also an interesting direction for additional experiments.

6.4 Coverage

In search of a suitable algorithm for monitoring an environment with multiple robots, literature research has revealed the coverage control approach as a fitting option. The main drawback of the approach has been identified: it only works in convex environments. Even though extensions to nonconvex environments already exist, they also had several disadvantages that made them not suitable for application in the envisioned setting.

Therefore a new approach that is less demanding when it comes to computation and communication has been developed. To this end, the visibility-limited Voronoi cells have been introduced and combined with the concept of contracting the environment or its boundaries. After a mathematical analysis and simplification of the objective function's gradient, a proof that all robot trajectories will stay inside the allowed environment was presented. Again, simulations verified the applicability of the presented approach and could be extended with more realistic models in upcoming works.

Future work could further focus on the comparison of approximated solutions with the exact solutions to the objective function to evaluate the difference in optimality and gain more insights into the properties of the presented method. Another possibility lies in the way the partitions are defined on visibility sets, and a formal proof of convergence to centroidal configurations is also open for future research.

6.5 Complete System Architecture

One of the goals stated in the beginning is to relieve operators from difficult control tasks and allow for operation in communication restricted settings by implementing a high level of autonomy. This goal has been achieved in several ways. Especially in the first stage, coordinating two robots manually in the desired manner would be difficult for someone who is not particularly trained. Coordinating multiple robots for exploration, especially in higher numbers, is also a delicate task for a single operator or even a team of operators. Additionally, the proposed pairwise optimization procedure is ideally suited for communication restricted settings because the communication only requires a minimum of two robots to start the target exchange.

A critical point for the overall system architecture to function is the transition and interconnection between the stages. Strictly speaking, stage one – even though it can be used stand-alone – should be active during the other two stages, i.e., during exploration and coverage. So far, the simplifying assumption to model one UAV-UGV pair as a single integrator for the other mission aspects has been used. How valid this assumption is can be further analyzed theoretically as well as experimentally. Another aspect of the simplification concerns the sensed area, which is assumed to be a fixed shape in stages two and three. When modeling the actual exploration units this area is dynamically changing and depends on the UAVs and UGVs movement. A more precise modeling will obviously influence the performance of the exploration and coverage and therefore provide new insights for further advancements.

Another interesting point for investigation is the transition between the second and third stage, i.e., how do the final positions at the end of the exploration influence the coverage performance? Is there a way to influence or improve these results?

To examine the complete system architecture, it seems natural to design an example mission and simulate all three aspects in one framework with the models from the first stage or even more realistic ones. More extensions that could be integrated include the actual victim detection, determination of density (i.e., areas of higher interest) or other application examples, depending on the type of task.

Real hardware experiments would require discrete control algorithms and communication in discrete time, which might add some further challenges and lead to unexpected experimental results. Unreliable connections with interruptions and data loss may cause further problems that could be addressed in future work.

As the conclusion shows, there are several practical aspects that still have to be considered before the methods and algorithms developed and presented in this thesis can be applied in a real search and rescue mission. The purpose was to gain additional insight on new or already existing ideas and bring them closer to a realization in a meaningful scenario. This goal has been reached successfully. The theoretical results provide important guarantees while still being relatively simple to implement in practical scenarios. Thus, the proposed approaches represent building blocks for future exploration systems with a particular focus on search and rescue missions with the prospect of a better performance compared to the previous techniques.

Bibliography

[1] Acevedo, J. J., Arrue, B. C., Maza, I., and Ollero, A. (2014). A decentralized algorithm for area surveillance missions using a team of aerial robots with different sensing capabilities. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4735–4740. IEEE.

[2] Adamy, J. (2014). *Nichtlineare Systeme und Regelungen.* Springer-Verlag.

[3] An, B. and Tambe, M. (2011). Game theory for security: an important challenge for multiagent systems. In *European Workshop on Multi-Agent Systems*, pages 17–30. Springer.

[4] Antonelli, G. (2013). Interconnected dynamic systems: An overview on distributed control. *IEEE Control Systems Magazine*, 33(1):76–88.

[5] Aristotle (1932). *Politics.* Harvard University Press.

[6] Bajcsy, R. (1988). Active perception. *Proceedings of the IEEE*, 76(8):966–1005.

[7] Barnes, D. and Gray, J. (1991). Behaviour synthesis for co-operant mobile robot control. In *International Conference on Control*, pages 1135–1140.

[8] Berhault, M., Huang, H., Keskinocak, P., Koenig, S., Elmaghraby, W., Griffin, P., and Kleywegt, A. (2003). Robot exploration with combinatorial auctions. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1957–1962. IEEE.

[9] Breitenmoser, A., Schwager, M., Metzger, J.-C., Siegwart, R., and Rus, D. (2010). Voronoi coverage of non-convex environments with a group of networked robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4982–4989. IEEE.

[10] Broekens, J., Heerink, M., and Rosendal, H. (2009). Assistive social robots in elderly care: a review. *Gerontechnology*, 8(2):94–103.

[11] Bullo, F., Cortés, J., and Martínez, S. (2009). *Distributed Control of Robotic Networks*. Princeton University Press.

[12] Burgard, W., Moors, M., Stachniss, C., and Schneider, F. (2005). Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3):376–386.

[13] Caicedo-Núñez, C. and Žefran, M. (2008a). A coverage algorithm for a class of non-convex regions. In *47th IEEE Conference on Decision and Control (CDC)*, pages 4244–4249. IEEE.

[14] Caicedo-Núñez, C. and Žefran, M. (2008b). Performing coverage on nonconvex domains. In *IEEE International Conference on Control Applications (CCA)*, pages 1019–1024. IEEE.

[15] Cantelli, L., Mangiameli, M., Melita, C., and Muscato, G. (2013). UAV/UGV cooperation for surveying operations in humanitarian demining. In *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–6.

[16] Cao, Y. U., Fukunaga, A. S., and Kahng, A. (1997). Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4(1):7–27.

[17] Capitan, J., Spaan, M. T., Merino, L., and Ollero, A. (2013). Decentralized multi-robot cooperation with auctioned POMDPs. *The International Journal of Robotics Research*, 32(6):650–671.

[18] Chaimowicz, L., Grocholsky, B., Keller, J. F., Kumar, V., and Taylor, C. J. (2004). Experiments in multirobot air-ground coordination. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4053–4058. IEEE.

[19] Chen, J., Gauci, M., and Groß, R. (2013). A strategy for transporting tall objects with a swarm of miniature mobile robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 863–869. IEEE.

[20] Choi, H.-L. and How, J. P. (2010). Continuous trajectory planning of mobile sensors for informative forecasting. *Automatica*, 46(8):1266–1275.

[21] Choi, J., Oh, S., and Horowitz, R. (2007). Cooperatively learning mobile agents for gradient climbing. In *IEEE Conference on Decision and Control (CDC)*, pages 3139–3144. IEEE.

[22] Choset, H. (2001). Coverage for robotics–a survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31(1-4):113–126.

[23] Cortés, J., Martínez, S., and Bullo, F. (2005). Spatially-distributed coverage optimization and control with limited-range interactions. *ESAIM: Control, Optimisation and Calculus of Variations*, 11:691–719.

[24] Cortés, J., Martínez, S., Karatas, T., and Bullo, F. (2004). Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255.

[25] Cruz, D., McClintock, J., Perteet, B., Orqueda, O., Cao, Y., and Fierro, R. (2007). Decentralized cooperative control-a multivehicle platform for research in networked embedded systems. *IEEE Control Systems Magazine*, 3(27):58–78.

[26] Dewan, A., Mahendran, A., Soni, N., and Krishna, K. M. (2013). Heterogeneous UGV-MAV exploration using integer programming. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5742–5749. IEEE.

[27] Dias, M. B., Zlot, R., Kalra, N., and Stentz, A. (2006). Market-based multirobot coordination: A survey and analysis. *Proceedings of the IEEE*, 94(7):1257–1270.

[28] Dirichlet, G. L. (1850). Über die Reduction der positiven quadratischen Formen mit drei unbestimmten ganzen Zahlen. *Journal für die reine und angewandte Mathematik*, 40:209–227.

[29] Du, Q., Faber, V., and Gunzburger, M. (1999). Centroidal voronoi tessellations: applications and algorithms. *SIAM Review*, 41(4):637–676.

[30] Durham, J. W., Carli, R., Frasca, P., and Bullo, F. (2012). Discrete partitioning and coverage control for gossiping robots. *IEEE Transactions on Robotics*, 28:364–378.

[31] Ehrenmann, M., Zollner, R., Rogalla, O., and Dillmann, R. (2002). Programming service tasks in household environments by human

demonstration. In *11th IEEE International Workshop on Robot and Human Interactive Communication*, pages 460–467. IEEE.

[32] Faigl, J. (2010). Approximate solution of the multiple watchman routes problem with restricted visibility range. *IEEE Transactions on Neural Networks*, 21(10):1668–1679.

[33] Faigl, J., Kulich, M., and Přeučil, L. (2012). Goal assignment using distance cost in multi-robot exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3741–3746. IEEE.

[34] Flanders, H. (1973). Differentiation under the integral sign. *The American Mathematical Monthly*, 80(6):615–627.

[35] Forster, C., Pizzoli, M., and Scaramuzza, D. (2013). Air-ground localization and map augmentation using monocular dense reconstruction. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3971–3978. IEEE.

[36] Fox, D., Ko, J., Konolige, K., Limketkai, B., Schulz, D., and Stewart, B. (2006). Distributed multirobot exploration and mapping. *Proceedings of the IEEE*, 94(7):1325–1339.

[37] Franchi, A., Oriolo, G., and Stegagno, P. (2010). Probabilistic mutual localization in multi-agent systems from anonymous position measures. In *49th IEEE Conference on Decision and Control (CDC)*, pages 6534–6540. IEEE.

[38] Galceran, E. and Carreras, M. (2013). A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12):1258–1276.

[39] Ganguli, A., Cortés, J., and Bullo, F. (2008). Distributed coverage of nonconvex environments. In *Networked Sensing Information and Control*, pages 289–305. Springer US.

[40] Garzón, M., Valente, J., Zapata, D., and Barrientos, A. (2013). An aerial-ground robotic system for navigation and obstacle mapping in large outdoor areas. *Sensors*, 13(1):1247–1267.

[41] Gazi, V. and Fidan, B. (2006). Coordination and control of multi-agent dynamic systems: Models and approaches. In *International Workshop on Swarm Robotics*, pages 71–102. Springer.

[42] Ge, S. S. and Fua, C.-H. (2005). Complete multi-robot coverage of unknown environments with minimum repeated coverage. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 715–720. IEEE.

[43] Gerkey, B., Vaughan, R. T., and Howard, A. (2003). The player/stage project: Tools for multi-robot and distributed sensor systems. In *Proceedings of the 11th International Conference on Advanced Robotics*, volume 1, pages 317–323.

[44] Gerkey, B. P. and Matarić, M. J. (2004). A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, 23(9):939–954.

[45] Golfarelli, M., Maio, D., and Rizzi, S. (1997). Multi-agent path planning based on task-swap negotiation. In *Proc. 16th UK Planning and Scheduling SIG Workshop*, pages 69–82.

[46] Grocholsky, B., Keller, J., Kumar, V., and Pappas, G. (2006). Cooperative air and ground surveillance. *IEEE Robotics & Automation Magazine*, 13(3):16–25.

[47] Haumann, D. (2015). *Distributed Multi-Robot Exploration.* PhD thesis, TU Darmstadt, Darmstadt. Tag der mündlichen Prüfung: 02.07. 2014.

[48] Haumann, D., Breitenmoser, A., Willert, V., Listmann, K., and Siegwart, R. (2011). DisCoverage for non-convex environments with arbitrary obstacles. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4486–4491. IEEE.

[49] Haumann, D., Willert, V., and Listmann, K. D. (2013). DisCoverage: from coverage to distributed multi-robot exploration. *IFAC Proceedings Volumes*, 46(27):328–335.

[50] Haumann, D., Willert, V., and Wahrburg, A. (2012). Kalman filtering in mobile consensus networks. In *International Symposium on Intelligent Control*, pages 944–950. IEEE.

[51] Heap, B. and Pagnucco, M. (2012). Analysis of cluster formation techniques for multi-robot task allocation using sequential single-cluster auctions. In *Australasian Conference on Artificial Intelligence*, pages 839–850. Springer.

[52] Heppner, G., Roennau, A., and Dillman, R. (2013). Enhancing sensor capabilities of walking robots through cooperative exploration with aerial robots. *Journal of Automation, Mobile Robotics & Intelligent Systems*, 7(2):5–11.

[53] Hoffmann, G. M., Huang, H., Waslander, S. L., and Tomlin, C. J. (2007). Quadrotor helicopter flight dynamics and control: Theory and experiment. In *Proc. of the AIAA Guidance, Navigation, and Control Conference*, volume 2, page 4.

[54] Hollinger, G., Singh, S., and Kehagias, A. (2010). Improving the efficiency of clearing with multi-agent teams. *The International Journal of Robotics Research*, 29(8):1088–1105.

[55] Homer (1924). *The Iliad.* Harvard University Press, 1st edition.

[56] Howard, A., Matarić, M. J., and Sukhatme, G. S. (2002). Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem. In Asama, H., Arai, T., Fukuda, T., and Hasegawa, T., editors, *Distributed Autonomous Robotic Systems 5*, pages 299–308. Springer Japan.

[57] Hurwitz, A. (1895). Ueber die Bedingungen, unter welchen eine Gleichung nur Wurzeln mit negativen reellen Theilen besitzt. *Mathematische Annalen*, 46(2):273–284.

[58] Hussein, I. I. and Stipanović, D. M. (2006). Effective coverage control using dynamic sensor networks. In *45th IEEE Conference on Decision and Control (CDC)*, pages 2747–2752. IEEE.

[59] Hussein, I. I. and Stipanović, D. M. (2007). Effective coverage control for mobile sensor networks with guaranteed collision avoidance. *IEEE Transactions on Control Systems Technology*, 15(4):642–657.

[60] Jadbabaie, A., Lin, J., and Morse, A. S. (2003). Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001.

[61] Ji, M. and Egerstedt, M. (2007). Distributed coordination control of multiagent systems while preserving connectedness. *IEEE Transactions on Robotics*, 23(4):693–703.

[62] Julian, B. J., Angermann, M., Schwager, M., and Rus, D. (2012). Distributed robotic sensor networks: An information-theoretic approach. *The International Journal of Robotics Research*, 31(10):1134–1154.

[63] Khalil, H. K. (2001). *Nonlinear Systems*. Pearson, 3rd edition.

[64] Khodaverdian, S. and Adamy, J. (2014). Robust output synchronization for heterogeneous multi-agent systems based on input-output decoupling. In *11th IEEE International Conference on Control & Automation (ICCA)*, pages 607–613. IEEE.

[65] Klodt, L., Haumann, D., and Willert, V. (2014). Revisiting coverage control in nonconvex environments using visibility sets. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 82–89. IEEE.

[66] Klodt, L., Khodaverdian, S., and Willert, V. (2015). Motion control for uav-ugv cooperation with visibility constraint. In *IEEE Conference on Control Applications (CCA)*, pages 1379–1385. IEEE.

[67] Klodt, L. and Willert, V. (2015). Equitable workload partitioning for multi-robot exploration through pairwise optimization. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2809–2816. IEEE.

[68] Kloetzer, M., Ding, X. C., and Belta, C. (2011). Multi-robot deployment from ltl specifications with reduced communication. In *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 4867–4872. IEEE.

[69] Knepper, R. A., Layton, T., Romanishin, J., and Rus, D. (2013). Ikeabot: An autonomous multi-robot coordinated furniture assembly system. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 855–862. IEEE.

[70] Koenig, N. and Howard, A. (2004). Design and use paradigms for gazebo, an open-source multi-robot simulator. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2149–2154. IEEE.

[71] Krajník, T., Nitsche, M., Faigl, J., Vaněk, P., Saska, M., Přeučil, L., Duckett, T., and Mejail, M. (2014). A practical multirobot localization system. *Journal of Intelligent & Robotic Systems*, 76(3-4):539–562.

[72] Kruskal, J. B. (1956). On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7(1):48–50.

[73] Kulich, M., Faigl, J., and Přeučil, L. (2011). On distance utility in the exploration task. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4455–4460. IEEE.

[74] Lagoudakis, M. G., Markakis, E., Kempe, D., Keskinocak, P., Kleywegt, A. J., Koenig, S., Tovey, C. A., Meyerson, A., and Jain, S. (2005). Auction-based multi-robot routing. In *Robotics: Science and Systems*, volume 5, pages 343–350.

[75] Langerwisch, M., Wittmann, T., Thamke, S., Remmersmann, T., Tiderko, A., and Wagner, B. (2013). Heterogeneous teams of unmanned ground and aerial robots for reconnaissance and surveillance-a field experiment. In *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–6.

[76] LaSalle, J. P. (1960). Some extensions of Liapunov's second method. *IRE Transactions on Circuit Theory*, 7(4):520–527.

[77] LaValle, S. M. (2011). Motion planning part i: The essentials. *IEEE Robotics & Automation Magazine*, 18(1):79–89.

[78] Laventall, K. and Cortés, J. (2009). Coverage control by multi-robot networks with limited-range anisotropic sensory. *International Journal of Control*, 82(6):1113–1121.

[79] Lemaire, T., Alami, R., and Lacroix, S. (2004). A distributed tasks allocation scheme in multi-uav context. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3622–3627. IEEE.

[80] Leyton-Brown, K. and Shoham, Y. (2008). Essentials of game theory: A concise multidisciplinary introduction. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2(1):1–88.

[81] Li, W., Zhang, T., and Kuhnlenz, K. (2011). A vision-guided autonomous quadrotor in an air-ground multi-robot system. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2980–2985. IEEE.

[82] Listmann, K. D., Wahrburg, A., Strubel, J., Adamy, J., and Konigorski, U. (2011). Partial-state synchronization of linear heterogeneous multi-agent systems. In *50th IEEE Conference on Decision and Control and European Control Conference (CDC-ECC)*, pages 3440–3445. IEEE.

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

[84] Lozano-Pérez, T. and Wesley, M. A. (1979). An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 22(10):560–570.

[85] Lu, L., Choi, Y.-K., and Wang, W. (2011). Visibility-based coverage of mobile sensors in non-convex domains. In *Eighth International Symposium on Voronoi Diagrams in Science and Engineering (ISVD)*, pages 105–111.

[86] Lyapunov, A. M. (1992). The general problem of the stability of motion. *International Journal of Control*, 55(3):531–534.

[87] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297.

[88] Maimone, M., Matthies, L., Osborn, J., Rollins, E., Teza, J., and Thayer, S. (1998). A photo-realistic 3-d mapping system for extreme nuclear environments: Chernobyl. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1521–1527. IEEE.

[89] Makarenko, A., Williams, S., Bourgault, F., and Durrant-Whyte, H. (2002). An experiment in integrated exploration. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 534–539. IEEE.

[90] Marier, J.-S., Rabbath, C.-A., and Léchevin, N. (2012). Visibility-limited coverage control using nonsmooth optimization. In *Proceedings of the American Control Conference (ACC)*, pages 6029–6034.

[91] Martínez, S., Cortés, J., and Bullo, F. (2007). Motion coordination with distributed information. *IEEE Control Systems*, 27(4):75–88.

[92] Maslow, A. H. (1943). A theory of human motivation. *Psychological Review*, 50(4):370.

[93] Mellinger, D., Kushleyev, A., and Kumar, V. (2012). Mixed-integer quadratic program trajectory generation for heterogeneous quadrotor teams. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 477–483. IEEE.

[94] Merino, L., Caballero, F., Martínez-de Dios, J. R., Ferruz, J., and Ollero, A. (2006). A cooperative perception system for multiple UAVs: Application to automatic detection of forest fires. *Journal of Field Robotics*, 23(3-4):165–184.

[95] Michael, N., Shen, S., Mohta, K., Mulgaonkar, Y., Kumar, V., Nagatani, K., Okada, Y., Kiribayashi, S., Otake, K., Yoshida, K., et al. (2012). Collaborative mapping of an earthquake-damaged building via ground and aerial robots. *Journal of Field Robotics*, 29(5):832–841.

[96] Miller, D. P., Nourbakhsh, I. R., and Siegwart, R. (2008). Robots for education. In *Springer Handbook of Robotics*, pages 1283–1301. Springer.

[97] Moors, M., Rohling, T., and Schulz, D. (2005). A probabilistic approach to coordinated multi-robot indoor surveillance. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3447–3452. IEEE.

[98] Mosteo, A. R. and Montano, L. (2007). Comparative experiments on optimization criteria and algorithms for auction based multi-robot task allocation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3345–3350. IEEE.

[99] Mueggler, E., Faessler, M., Fontana, F., and Scaramuzza, D. (2014). Aerial-guided navigation of a ground robot among movable obstacles. In *IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–8.

[100] Murray, R. M. (2007). Recent research in cooperative control of multivehicle systems. *Journal of Dynamic Systems, Measurement, and Control*, 129(5):571–583.

[101] Nistér, D., Naroditsky, O., and Bergen, J. (2004). Visual odometry. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–659. IEEE.

[102] Obermeyer, K. J., Ganguli, A., and Bullo, F. (2011). Multi-agent deployment for visibility coverage in polygonal environments with holes. *International Journal of Robust and Nonlinear Control*, 21(12):1467–1492.

[103] Ögren, P., Egerstedt, M., and Hu, X. (2001). A control lyapunov function approach to multi-agent coordination. In *40th IEEE Conference on Decision and Control (CDC)*, pages 1150–1155. IEEE.

[104] Oh, S., Schenato, L., Chen, P., and Sastry, S. (2007). Tracking and coordination of multiple agents using sensor networks: system design, algorithms and experiments. *Proceedings of the IEEE*, 95(1):234–254.

[105] Okabe, A. and Suzuki, A. (1997). Locational optimization problems solved through voronoi diagrams. *European Journal of Operational Research*, 98(3):445–456.

[106] Olfati-Saber, R. (2006). Flocking for multi-agent dynamic systems: Algorithms and theory. *IEEE Transactions on Automatic Control*, 51(3):401–420.

[107] Olfati-Saber, R., Fax, J. A., and Murray, R. M. (2007). Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233.

[108] Olfati-Saber, R. and Murray, R. M. (2004). Consensus problems in networks of agents with switching topology and time-delays. *IEEE Transactions on Automatic Control*, 49(9):1520–1533.

[109] Parker, L. E. (1998). Alliance: An architecture for fault tolerant multirobot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240.

[110] Parker, L. E. (2008). Multiple mobile robot systems. In *Springer Handbook of Robotics*, pages 921–941. Springer.

[111] Pasqualetti, F., Franchi, A., and Bullo, F. (2010). On optimal cooperative patrolling. In *49th IEEE Conference on Decision and Control (CDC)*, pages 7153–7158. IEEE.

[112] Pimenta, L. C., Schwager, M., Lindsey, Q., Kumar, V., Rus, D., Mesquita, R. C., and Pereira, G. A. (2009). Simultaneous coverage and tracking (SCAT) of moving targets with robot networks. In *Algorithmic Foundation of Robotics VIII*, pages 85–99. Springer.

[113] Pimenta, L. C. A., Kumar, V., Mesquita, R., and Pereira, G. A. S. (2008). Sensing and coverage for a network of heterogeneous robots. In *47th IEEE Conference on Decision and Control (CDC)*, pages 3947–3952. IEEE.

[114] Powers, C., Mellinger, D., and Kumar, V. (2015). Quadrotor kinematics and dynamics. In *Handbook of Unmanned Aerial Vehicles*, pages 307–328. Springer.

[115] Prassler, E., Ritter, A., Schaeffer, C., and Fiorini, P. (2000). A short history of cleaning robots. *Autonomous Robots*, 9(3):211–226.

[116] Prim, R. C. (1957). Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36(6):1389–1401.

[117] Putz, P. (1998). Space robotics in europe: A survey. *Robotics and Autonomous Systems*, 23(1):3–16.

[118] Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). ROS: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, volume 3. Kobe, Japan.

[119] Rekleitis, I., New, A. P., Rankin, E. S., and Choset, H. (2008). Efficient boustrophedon multi-robot coverage: an algorithmic approach. *Annals of Mathematics and Artificial Intelligence*, 52(2-4):109–142.

[120] Rekleitis, I. M., Dudek, G., and Milios, E. E. (2002). Multi-robot cooperative localization: a study of trade-offs between efficiency and accuracy. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2690–2695. IEEE.

[121] Ren, W., Beard, R. W., and Atkins, E. M. (2007). Information consensus in multivehicle cooperative control. *IEEE Control Systems Magazine*, 27(2):71–82.

[122] Renzaglia, A. and Martinelli, A. (2009). 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*. IEEE.

[123] Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *ACM SIGGRAPH Computer Graphics*, 21(4):25–34.

[124] Roberts, A. (2006). *The History of Science Fiction.* Palgrave Histories of Literature. Palgrave Macmillan UK, 1st edition.

[125] Rohmer, E., Singh, S. P., and Freese, M. (2013). V-REP: A versatile and scalable robot simulation framework. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1321–1326. IEEE.

[126] Rosenkrantz, D. J., Stearns, R. E., and Lewis, P. M. (1977). An analysis of several heuristics for the traveling salesman problem. *SIAM Journal on Computing*, 6:563–581.

[127] Routh, E. J. (1877). *A treatise on the stability of a given state of motion: particularly steady motion.* Macmillan and Company.

[128] Salih, A. L., Moghavvemi, M., Mohamed, H. A., and Gaeid, K. S. (2010). Modelling and pid controller design for a quadrotor unmanned air vehicle. In *IEEE International Conference on Automation Quality and Testing Robotics (AQTR)*, pages 1–5. IEEE.

[129] Sandholm, T. (1998). Contract types for satisficing task allocation. In *Proceedings of the AAAI spring symposium: Satisficing models*, pages 23–25.

[130] Saska, M., Vonásek, V., Krajník, T., and Přeučil, L. (2014). Coordination and navigation of heterogeneous MAV–UGV formations localized by a hawk-eye-like approach under a model predictive control scheme. *The International Journal of Robotics Research*, 33(10):1393–1412.

[131] Schwager, M., Dames, P., Rus, D., and Kumar, V. (2017). A multi-robot control policy for information gathering in the presence of unknown hazards. In *Robotics Research*, pages 455–472. Springer.

[132] Schwager, M., Mclurkin, J., and Rus, D. (2006). Distributed coverage control with sensory feedback for networked robots. In *Robotics: Science and Systems.*

[133] Semsar-Kazerooni, E. and Khorasani, K. (2009). Multi-agent team cooperation: A game theory approach. *Automatica*, 45(10):2205–2213.

[134] Seyboth, G. S., Dimarogonas, D. V., and Johansson, K. H. (2013). Event-based broadcasting for multi-agent average consensus. *Automatica*, 49(1):245–252.

[135] Shkurti, F., Xu, A., Meghjani, M., Higuera, J. C. G., Girdhar, Y., Giguere, P., Dey, B. B., Li, J., Kalmbach, A., Prahacs, C., et al. (2012). Multi-domain monitoring of marine environments using a heterogeneous robot team. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1747–1753. IEEE.

[136] Singh, A., Krause, A., and Kaiser, W. J. (2009). Nonmyopic adaptive informative path planning for multiple robots. In *Proceedings of the 21st International Joint Conference on Artifical Intelligence*, pages 1843–1850. Morgan Kaufmann Publishers Inc.

[137] Smith, R. (1980). The contract net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers*, 29(12):1104–1113.

[138] Solanas, A. and Garcia, M. (2004). Coordinated multi-robot exploration through unsupervised clustering of unknown space. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 717–721. IEEE.

[139] Steinhaus, H. (1956). Sur la division des corp materiels en parties. *Bull. Acad. Polon. Sci*, 1:801–804.

[140] Sujit, P. and Beard, R. (2007). Cooperative path planning for multiple uavs exploring an unknown region. In *American Control Conference (ACC)*, pages 347–352. IEEE.

[141] Tanner, H. and Christodoulakis, D. (2006). Cooperation between aerial and ground vehicle groups for reconnaissance missions. In *IEEE Conference on Decision and Control (CDC)*, pages 5918–5923. IEEE.

[142] Tanner, H. G., Pappas, G. J., and Kumar, V. (2004). Leader-to-formation stability. *IEEE Transactions on Robotics and Automation*, 20(3):443–455.

[143] Valero-Gomez, A., Valero-Gomez, J., Castro-Gonzalez, A., and Moreno, L. (2011). Use of genetic algorithms for target distribution and sequencing in multiple robot operations. In *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 2718–2724. IEEE.

[144] Voronoi, G. F. (1907). Nouvelles applications des parametres continus a la théorie des formes quadratiques. *Journal für die reine und angewandte Mathematik*, 133:97–178.

[145] Wang, J., Wu, L., Meng, M. Q.-H., and Ren, H. (2014). Towards simultaneous coordinate calibrations for cooperative multiple robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 410–415. IEEE.

[146] Wang, X., Yadav, V., and Balakrishnan, S. (2007). Cooperative uav formation flying with obstacle/collision avoidance. *IEEE Transactions on Control Systems Technology*, 15(4):672–679.

[147] West, D. B. et al. (2001). *Introduction to graph theory*, volume 2. Prentice Hall.

[148] Wu, L., García, M. A., Puig, D., and Sole, A. (2007). Voronoi-based space partitioning for coordinated multi-robot exploration. *Journal of Physical Agents*, 1(1):37–44.

[149] Wurm, K. M., Stachniss, C., and Burgard, W. (2008). Coordinated multi-robot exploration using a segmentation of the environment. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1160–1165. IEEE.

[150] Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 146–151.

[151] Yan, Z., Jouandeau, N., and Cherif, A. A. (2013). A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems*, 10:399–416.

[152] Zheng, X. and Koenig, S. (2009). K-swaps: cooperative negotiation for solving task-allocation problems. In *Proceedings of the 21st International Joint Conference on Artifical intelligence*, pages 373–378.

[153] Zhong, M. and Cassandras, C. G. (2008). Distributed coverage control in sensor network environments with polygonal obstacles. *IFAC Proceedings Volumes*, 41(2):4162–4167.

[154] Zimmermann, J. (2008). *Adaptive Multi-Agenten-Systeme zur Steuerung komplexer Produktionssysteme.* PhD thesis, Fernuniversität Hagen.

[155] Zlot, R., Stentz, A., Dias, M. B., and Thayer, S. (2002). Multi-robot exploration controlled by a market economy. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3016–3023. IEEE.

Publications and Supervised Theses

Publications [65–67] have been developed in the course of the dissertation. Further, the following student projects have been supervised during the time at the institute (in chronological order):

Type	**Topic**
Projektseminar	Pitchregelung einer Windturbine
Projektseminar	Bestimmung des nächsten Zielpunktes für Roboter-Exploration
Bachelorthesis	Strategies for obstacle avoidance in a non-convex environment
Bachelorthesis	Funktionsschätzer für Gebietsabdeckung mit mobilen Sensornetzwerken
Bachelorthesis	Lokalisierung und 3D-Kartierung mittels RGB-D Kamera für mobile Roboter
Masterthesis	Ein heuristischer Ansatz zur sensorbasierten Multi-Robot Exploration mittels spiralförmiger, flächendeckender Laufbahnplanung
Bachelorthesis	Koordinierte Multi-Roboter Exploration mittels Consensus
Projektseminar	Implementierung und Analyse verschiedener Bewegungsdynamiken für Roboter-Simulationen
Masterthesis	Kartierung mittels Kinect in einem Multi-Roboter-System
Masterthesis	Coordination of Heterogeneous Multi-Robot Teams
Bachelorthesis	Realistische Simulation einer Roboter-Erkundung

Type	**Topic**
Studienarbeit	Bewegungsplanung für das aktive Messen räumlich verteilter Felder
Bachelorthesis	Kartierung mit einem Multiagentensystem
Studienarbeit	Kinematic calibration of a lightweight robot arm using narrow-band ultrasonic sensors
Projektseminar	Koordinierte Roboter-Exploration mit eingeschränkter Kommunikationsreichweite
Studienarbeit	Bewegungsregelungen und kürzeste Pfadplanung für ein UAV-UGV-System mit Erkundungsauftrag
Bachelorthesis	Sampling-basierte Roboter Exploration
Projektseminar	Entwurf und Simulation eines Robotischen Rüsseltiers in V-REP

Index

active sensing, 7
attractivity, 17
automaton, automata, 1

center of mass, 68
centralized coordination, 42
centroid, 68
centroidal Voronoi config., 70
characteristic polynomial, 19
circular motion, 36, 37
communication range, 52
communication round, 52
configuration, 69
consensus, 5
contract net protocol, 44
convex, 15
cooperation, 3
coverage and tracking, 32
coverage control, 62
coverage path planning, 6

decentralized system, 42
Delaunay graph, 67
delta-contraction, 73
density function, 24, 69
dynamic control law, 30
dynamic coverage, 24, 40
dynamical system, 16

edges, 13
effective coverage, 23
environment, 15
equilibrium partition, 52
equilibrium point, 17
error dynamics, 30
error function, 23
exploration framework, 54
exploration unit, 46

flocking, 5
formation, 6
frontier-based exploration, 41

generator points, 67
geodesic distance, 16, 47
goal selection, 41
graph theory, 13

invariant set, 18

k-means clustering, 43, 58

LaSalle's invariance principle, 18
linear time invariant system, 19
Lloyd algorithm, 67
localization, 8
Lyapunov function, 18
Lyapunov's direct method, 18

maneuverable domain, 47, 72
minimum spanning tree, 14, 48
minimum spanning trees, 45
multi-agent system, 3
multi-robot exploration, 41
multi-robot routing, 44

multi-robot task allocation, 44

nonconvex, 15

pairwise optimization, 49
partitioning, 43
patrolling, 7
performance function, 69
polygon, 15
position error, 30

relative localization, 20
robot, robota, 1
Routh-Hurwitz criterion, 19

search and rescue, 3, 10
sensor model, 22
set partitioning problem, 50
simulations, 5
spanning tree, 13
spatial estimation, 7
stability, 17
state feedback controller, 30
surveillance, 7
switching strategy, 34

task allocation, 43
time-varying density, 26
tracking control, 29
tracking error, 30, 36
transfer matrix, 19, 31
traveling salesman, 41

vertices, 13
virtual point tracking, 34
visibility, 16
visibility constraint, 24
visibility graph, 55
visibility-limited Voronoi cell, 72
Voronoi coverage, 63
Voronoi partition, 44, 66

workload, 48
workload balancing, 51

Lebenslauf

Persönliche Daten	
Name	Lukas Klodt
Anschrift	Steinackerstr. 96 64372 Ober-Ramstadt
Geburtsdatum	20. 09. 1986
Geburtsort	Rotenburg an der Fulda, Hessen

Werdegang	
1993 – 2006	Grundschule, Gesamtschule, Oberstufe und Abitur in Rotenburg an der Fulda
07/2006 – 03/2007	Zivildienst, Jugendherberge, Rotenburg an der Fulda
09/2007 – 03/2011	Bachelorstudium Elektrotechnik und Informationstechnik, Darmstadt
08/2011 – 04/2013	Masterstudium Elektrotechnik und Informationstechnik, Darmstadt
05/2013 – 05/2016	DFG Promotionsstipendium im Graduiertenkolleg 1362, TU Darmstadt
05/2016 – 11/2016	Promotionsstipendium der TU Darmstadt