UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

University of Modena and Reggio Emilia

Department of Sciences and Methods for Engineering

Doctoral school of Industrial Innovation Engineering

XXXVIII cycle

Control and Coordination of Multi-Robot Systems: From Theory to Field-Tested Deployments

---

Author:
Mehdi Belal

Supervisors:
Prof. Lorenzo Sabattini
Dr. Dario Albani

PhD Coordinator:
Prof. Franco Zambonelli

March 2026

Reggio Emilia, Italy, March 2026

To my Family

Acknowledgements

I would like to first thank my Family for their unconditional support and constant encouragement throughout this journey. Without them, this achievement would not have been possible.

I am deeply grateful to my advisor, Prof. Lorenzo Sabattini, for offering me the opportunity to start this PhD journey when it was least expected, and for his guidance, support, and patience over the years.

I sincerely thank Dr. Dario Albani, my co-advisor, for his continuous support and for providing new challenges and research directions that significantly enriched this work.

I also thank the Technology Innovation Institute and all the people I had the pleasure to work with for making it possible to pursue this PhD alongside my professional activity and for their support throughout the journey.

Abstract

Multi-robot systems are rapidly evolving and, thanks to technological advances, are expected to become an integral part of everyday life. Compared to single robots, fleets offer greater efficiency, flexibility, and fault tolerance. However, like any maturing technology, their management, control, and recovery from error conditions pose complex challenges. To fully exploit these technologies, it is essential to develop advanced and robust control strategies at both the individual and collective levels. The goal is to improve key parameters such as productivity, accuracy, safety, and cost efficiency through the coordination of multiple robots. This coordination relies on essential capabilities such as perception, action, communication, learning, and control.

This research focuses on new strategies to make multi-robot systems more effective and applicable in real-world contexts. In particular, we examine distributed control methods that do not require a centralized unit, enabling each robot to make autonomous decisions to optimize overall performance and allowing an operator to interact with the entire system rather than with individual agents.

One crucial aspect concerns optimizing the coverage of unknown environments for monitoring and exploration, especially in settings that are inaccessible or hazardous to humans. Traditional coverage control assumes complete knowledge of the environment and the robotic system—an assumption that is often unrealistic. To overcome this limitation, we developed a distributed control approach that accounts for communication constraints among vehicles and with the operator.

Another central theme is the robots' ability to adapt to spatially varying phenomena, detected either directly or through other components of the system. Examples include ambient temperature, target identification, or external data integrated into the system. We propose a method to estimate these phenomena and to modify the system's behavior dynamically. The model is then extended to scenarios in which the phenomena evolve over time, moving beyond the assumption of static fields and enabling dynamic resource management.

Beyond ground-based exploration, we extend the approach to the analysis and monitor-

ing of objects and environments with vertical extent. This study draws inspiration from scanning techniques, structural monitoring, and the 3D reconstruction of elements using specialized sensors. Here again, agent behaviors can be adapted based on the collected information, enhancing operational effectiveness.

The proposed strategies make multi-robot systems more reliable and better suited to real-world scenarios, improving their adaptability and operability in dynamic, complex environments. Moreover, experimental work on real vehicles has helped refine these solutions, ensuring greater efficiency and practicality.

Sommario

I sistemi multi-robot stanno evolvendo rapidamente e, grazie ai progressi tecnologici, si prevede che diventeranno parte integrante della vita quotidiana. Rispetto ai robot singoli, le flotte offrono maggiore efficienza, flessibilità e resistenza ai guasti. Tuttavia, come ogni tecnologia in crescita, la loro gestione, il controllo e il recupero da situazioni di errori pongono sfide complesse.

Per sfruttare appieno queste tecnologie, è fondamentale sviluppare strategie di controllo avanzate e robuste, sia a livello individuale che collettivo. L'obiettivo è migliorare parametri chiave come produttività, precisione, sicurezza ed efficienza economica attraverso il coordinamento di più robot. Questo coordinamento si basa su funzionalità essenziali come percezione, azione, comunicazione, apprendimento e controllo.

Questa ricerca si concentra su nuove strategie per rendere i sistemi multi-robot più efficaci e applicabili in contesti reali. In particolare, esaminiamo metodi di controllo distribuito che non richiedono un'unità centralizzata, permettendo a ogni robot di prendere decisioni autonome per ottimizzare le prestazioni complessive e consentendo a un operatore di interagire con l'intero sistema piuttosto che con singoli agenti.

Uno degli aspetti cruciali riguarda l'ottimizzazione della copertura di ambienti sconosciuti per il monitoraggio e l'esplorazione, in particolare in contesti inaccessibili o pericolosi per l'uomo. Il tradizionale "Coverage Control" presuppone una conoscenza completa dell'ambiente e del sistema robotico, un'ipotesi spesso irrealistica. Per superare questa limitazione, abbiamo sviluppato un approccio di controllo distribuito che tiene conto delle restrizioni di comunicazione tra i veicoli e con l'operatore.

Un altro tema centrale riguarda la capacità dei robot di adattarsi a fenomeni spaziali variabili, rilevati direttamente o attraverso altre componenti del sistema. Esempi includono la temperatura ambientale, l'identificazione di obiettivi o dati esterni integrati nel sistema. Abbiamo proposto un metodo per stimare questi fenomeni e modificare dinamicamente il comportamento del sistema. Successivamente, il modello è stato esteso a scenari in cui i fenomeni evolvono nel tempo, superando l'ipotesi di staticità e consentendo una gestione dinamica delle risorse.

Oltre all'esplorazione terrestre, abbiamo ampliato l'approccio all'analisi e al monitoraggio di oggetti e ambienti con estensione verticale. Questo studio si ispira alle tecniche di scansione, monitoraggio di strutture e ricostruzione 3D di elementi mediante sensori specifici. Anche in questo caso, il comportamento degli agenti può essere adattato sulla base delle informazioni raccolte, migliorando l'efficacia operativa.

Le strategie proposte rendono i sistemi multi-robot più affidabili e adatti a scenari reali, migliorando la loro capacità di adattamento e operatività in ambienti dinamici e complessi. Inoltre, il lavoro sperimentale su veicoli reali ha permesso di affinare le soluzioni, garantendone maggiore efficienza e praticità.

Contents

List of Figures

List of Abbreviations

BVLoS Beyond Visual Line of Sight.

CVT Centroidal Voronoi Tessellation.

GPS Global Positioning System.

LLC Low Level Controller.

LoS Line of Sight.

MRS Multi Robot Sytem.

PDF Probability Density Function.

ROS Robot Operating System.

SITL Software In The Loop.

TII Technology Innovation Institute.

UAV Unmanned Aerial Vehicles.

List of Publications

Conference papers

- *F. Bertoncelli, M. Belal, D. Albani, F. Pratissoli, and L. Sabattini, "On limited-range coverage control for large-scale teams of aerial drones: Deployment and study," in* The 16th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2022*
- *M. Belal, D. Albani, and L. Sabattini, "Understanding the role of time-varying targets in adaptive distributed area coverage control," in* International Symposium on Experimental Robotics. *Springer, 2023, pp. 239–249*
- *M. Catellani, M. Belal, and L. Sabattini, "Dual-layer control architecture for cooperative environmental monitoring in multi-robot systems," in* The 17th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2024*
- *M. Belal, T. Manoni, D. Albani, and L. Sabattini, "Decentralized multi-robot coverage of hemispherical surfaces via fortune-based partitioning," in* Submitted to the 15th International Conference on Swarm Intelligence (ANTS)*, 2026*
- *——, "Experimental evaluation of decentralized hemispherical coverage for multi-uav systems," in* Proceedings of The International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2026*

Chapter 1

Introduction

1.1 Introduction

In recent decades, the deployment of multiple cooperating robots has emerged as a powerful paradigm for addressing complex real-world problems that are otherwise intractable or unsafe for human intervention. From disaster response to precision agriculture, from environmental monitoring to warehouse logistics, Multi Robot Sytem (MRS) are redefining how autonomous agents interact with, perceive, and manipulate the environment around them. The fundamental advantage of multi-robot systems lies in their ability to operate in a decentralized and distributed fashion. Unlike monolithic robotic systems, which may suffer from single points of failure or scalability bottlenecks, a team of robots can coordinate to perform tasks more efficiently, with greater robustness and adaptability. Distributed robotic algorithms allow agents to make local decisions based on partial information while contributing to a global objective, enabling scalability, fault tolerance, and flexibility in dynamic environments.

Disaster scenarios offer some of the most compelling motivations for multi-robot systems. In situations such as chemical spills, collapsed buildings, or nuclear accidents, human access is limited or extremely dangerous. Autonomous ground or aerial robots, equipped with specialized sensors and coordinated through distributed algorithms, can perform reconnaissance, map hazardous areas, detect survivors, or carry out manipulation tasks: their collaboration reduces human risk while accelerating mission outcomes. In 2011, a massive earthquake and tsunami caused a catastrophic failure at the Fukushima Daiichi Nuclear Power Plant, in Japan, resulting in the release of radioactive material: a multi-robot system became fundamental to deal with this nuclear disaster to solve the cleanup of the radioactive area [7]. Similarly, during Hurricane Harvey in 2017, public safety agencies in the United States coordinated large fleets of drones for flood monitoring, victim localization, and critical infrastructure inspection, marking one of the largest drone-assisted emergency responses to date [8].

Agricultural robotics is another domain experiencing transformative impact through distributed multi-robot coordination. Teams of autonomous agents are now capable of collectively managing large-scale farmland, performing tasks such as soil monitoring, crop

spraying, irrigation control, and selective harvesting. The distributed nature of these systems ensures adaptability to changing environmental conditions, resilience to partial failures, and significant reductions in resource usage—contributing toward sustainable and scalable food production. In the literature, numerous works have focused on the use of distributed systems to optimize agricultural operations, motivated by the large-scale nature of the environments in which such tasks are typically conducted [9, 10, 11].

Healthcare environments have also witnessed a growing integration of multi-robot solutions. Robots operating in hospital facilities can autonomously disinfect surfaces, deliver medication, assist in logistics, and monitor patient vitals. The distributed approach allows for seamless coordination across wards and departments, increasing throughput and reliability while reducing the burden on human staff. Particularly in pandemic or high-risk scenarios, this capability becomes crucial for maintaining operational continuity while ensuring patient and worker safety [12].

In parallel with technological progress in aerial robotics, regulatory frameworks have played a pivotal role in defining the conditions under which Unmanned Aerial Vehicles (UAV) can be deployed. Historically, regulations imposed by civil aviation authorities—such as the Federal Aviation Administration (FAA) in the United States and the European Union Aviation Safety Agency (EASA) in Europe—have been extremely conservative on the matter. Restrictions have included strict Line of Sight (LoS) operation, geofencing, altitude limits, and prohibitions on swarm deployment or autonomous flight in populated areas. These constraints, while justified by safety and airspace management considerations, have limited the operational scope of UAV-based multi-robot systems. Nonetheless, recent years have seen a progressive shift toward more permissive and structured regulatory environments. Regulatory initiatives such as the FAA's Part 107 rules and EASA's U-space framework [13, 14] provide guidelines for Beyond Visual Line of Sight (BVLoS) operations, risk-based mission categorization, and authorization pathways for complex autonomous operations. These developments reflect a growing consensus that aerial autonomy will play a central role in the future of mobility, logistics, and inspection. As a result, the path to real-world deployment of UAV swarms is gradually becoming clearer and more accessible.

Despite these advances, the effective coordination of a multi-robot system remains a central scientific and engineering challenge, especially in those situations outside the laboratory where high quality and reliable systems are required; the deployment of autonomous aerial vehicles poses significant challenges in terms of sensing, control, and coordination [15]. At the heart of these challenges lie a series of algorithmic problems: how to plan and allocate tasks across the team, how to maintain a shared understanding of the environment under communication constraints, and how to optimize coverage, formation, or exploration goals while ensuring safety and convergence guarantees, considering also time, and resources constrains—like battery usage among the others. These problems are compounded in environments where sensing is limited, the geometry of the workspace is non-trivial (e.g., convex or non-convex domains), or where prioritization of certain regions (e.g., based on probability distributions) is required.

This thesis work is situated at the intersection of control theory, distributed optimiza-

tion, and computational geometry. It investigates algorithmic frameworks and coordination strategies for distributed coverage, formation, and monitoring tasks in multi-robot systems. Emphasis is placed on algorithms that are provably correct, computationally tractable, and compatible with real-world constraints such as limited communication, sensing range, and dynamic environments. Techniques based on Voronoi tessellations, Lloyd's algorithm, Gaussian-weighted optimization, and proximity graph structures are developed and analyzed both theoretically and empirically. All proposed methodologies and theoretical results are further validated through extensive real-world experiments with multi-robot aerial platforms, ensuring their practical feasibility and robustness beyond simulation. The overarching goal is to contribute scalable, robust, and implementable solutions to the coordination of autonomous multi-agent systems, enabling their deployment in increasingly challenging and impactful application domains. The algorithms and methodologies presented in this thesis are designed with these evolving constraints in mind. Particular attention is given to ensuring that the coordination strategies are compatible with operational restrictions, scalable to larger teams, and suitable for integration into safety-critical environments. In this thesis, the coverage problems under consideration are defined on domains that are intrinsically two-dimensional, even when embedded in three-dimensional space. This includes both planar environments and non-planar surfaces, such as the hemisphere, for which the proposed methodologies extend two-dimensional coverage and partitioning concepts to geometrically three-dimensional settings.

1.2 Contribution and Thesis Outline

Multi-robot aerial systems have the potential to transform monitoring, exploration, and inspection tasks, but their deployment outside controlled laboratory conditions is still limited by several key challenges. Among these are the reliance on global information and idealized sensing, the difficulty of scaling coordination strategies to large teams, the complexity of operating in time-varying environments with points of interest, and the need to extend coverage methodologies beyond planar domains while preserving safety and robustness.

The main contribution of this work lies in the development, analysis, and experimental validation of distributed coordination strategies for multi-robot aerial systems, with a particular focus on coverage control in realistic scenarios. The work aims at closing the gap between theoretical research and practical deployment by:

- Designing coverage-based control strategies that rely only on local sensing and limited communication, and that scale to large teams of aerial robots;
- Incorporating time-varying phenomena, points of interest, and safety constraints (such as collision avoidance and formation maintenance) into the coverage framework;
- Extending coverage methodologies to non-planar, three-dimensional domains, with a specific focus on hemispherical environments and their algorithmic complexity;
- Rigorously validating the proposed approaches through realistic simulations and

real-world experiments with fleets of UAV.

The technical Chapters of this thesis should not be mwnt as isolated contributions, but as successive extensions of a common distributed coverage-control framework for multi-UAV systems. Starting from scalable limited-range coverage in planar environments, the thesis progressively incorporates dynamic targets and safety constraints, and finally generalizes the framework to non-planar hemispherical domains.

The dissertation is organized as follows. Chapter 2 provides the background necessary for a proper understanding of the work, introducing the coverage methodologies for multi-drone systems that form the foundation of this thesis. We first present the control and modeling framework adopted for aerial vehicles, including the motion control schemes. We then review coverage control concepts for UAV teams under limited sensing and communication, establishing the theoretical and methodological basis for the subsequent chapters.

Chapter 3 focuses on coverage control for large-scale UAV teams under limited sensing and communication. We consider a limited-range setting in which each vehicle can only access local information. We develop and analyze a coverage strategy based on limited Voronoi partitioning and demonstrate its scalability through extensive simulations and real-world experiments with a large fleet of quadrotor drones. The field trials, conducted in unstructured outdoor environments, highlight the feasibility of deploying fully distributed coverage controllers in realistic conditions. The study of this topic has produced the following publication:

- **On Limited-Range Coverage Control for Large-Scale Teams of Aerial Drones: Deployment and Study**
  Filippo Bertoncelli, Mehdi Belal, Dario Albani, Federico Pratissoli, Lorenzo Sabattini
  The 16th International Symposium on Distributed Autonomous Robotic Systems (DARS), 2022

Chapter 4 is devoted to the role of time variation and points of interest in distributed coverage control. We move beyond static coverage formulations and consider scenarios where regions of interest evolve over time, and where the team must simultaneously address formation requirements and safety constraints. We introduce mathematical tools to incorporate realistic complexity into the coverage problem, including time-varying density functions, Gaussian probability density functions (PDFs) to model points of interest, and control-theoretic mechanisms for collision avoidance and formation maintenance. In particular, we employ Control Barrier Functions (CBFs) to guarantee safety with respect to the environment and inter-robot collisions, and Control Lyapunov Functions (CLFs) to enforce formation-related objectives. The proposed controllers are validated through simulation and experimental campaigns, demonstrating their effectiveness in adaptive area coverage with time-varying targets and cooperative environmental monitoring. The study of these topics has produced the following works:

- **Understanding the role of time-varying targets in adaptive distributed area coverage control**
  Mehdi Belal, Dario Albani, Lorenzo Sabattini

The International Symposium on Experimental Robotics (ISER), 2023.

- **Dual-layer control architecture for cooperative environmental monitoring in multi-robot systems**
  Mattia Catellani, Mehdi Belal, Lorenzo Sabattini
  The 17th International Symposium on Distributed Autonomous Robotic Systems (DARS), 2024.

In Chapter 5 we address the scalability of coverage control to 2.5D and 3D domains, focusing in particular on hemispherical environments. Many aerial applications naturally involve non-planar target regions, such as dome-shaped observation spaces or spherical shells surrounding a point of interest. To bridge the gap between planar coverage and fully three-dimensional coordination, we extend coverage methodologies to curved surfaces and analyze their algorithmic and geometric implications. We develop a decentralized coverage framework for hemispherical domains that leverages geometric relationships on the sphere and adapts classical Voronoi-based partitioning to non-planar manifolds. In particular, we introduce a spherical variant of Fortune's sweepline algorithm, enabling distributed and computationally efficient partitioning on hemispherical surfaces. The resulting control strategy allows UAV teams to maintain coverage while navigating on a curved domain, supporting both uniform and density-weighted configurations. Special attention is given to points of interest, Gaussian density functions for non-axial formations, and the overall complexity of the partitioning and control pipeline. The proposed methodologies are validated through simulations and real-world experiments, demonstrating scalability with respect to team size. This Chapter contributes to the generalization of distributed coverage from planar environments to realistic 3D aerial scenarios. The study of this topic has produced the following works:

- **Decentralized multi-robot coverage of hemispherical surfaces via Fortune-based partitioning**
  Mehdi Belal, Tiziano Manoni, Dario Albani, Lorenzo Sabattini
  The 15th International Conference on Swarm Intelligence (ANTS), 2026.
- **Experimental Evaluation of Decentralized Hemispherical Coverage for Multi-UAV Systems**
  Mehdi Belal, Tiziano Manoni, Dario Albani, Lorenzo Sabattini
  Submitted to The 18th International Symposium on Distributed Autonomous Robotic Systems (DARS), 2026.

Chapter 6 collects the concluding remarks of the dissertation. We summarize the main contributions of the work, highlighting how the proposed coverage-based strategies address key challenges related to limited sensing, time-varying environments, safety constraints, and non-planar domains in multi-robot aerial systems. We also discuss the lessons learned from the experimental validation campaigns and outline several directions for future research, including tighter integration with learning-based components, further extensions to complex three-dimensional environments, and the deployment of the proposed methodologies in industrial or mission-critical applications.

Chapter 2

Coverage Control: Background

In this Chapter, we explore one of the foundational strategies for coordinating multiple autonomous robots: the coverage-based control framework.

In the coming years, large-scale deployment of autonomous systems—such as self-driving vehicles, aerial drones, underwater robots, and others—is expected to become increasingly prevalent in practical scenarios, tackling complex and demanding missions [16]. Multi-robot systems find application in diverse domains including precision farming, disaster response, environmental data collection, exploration, surveillance, search and rescue missions, and outdoor industrial inspection and diagnostics [17]. Distributed coordination architectures have long been studied as a means to achieve scalable and robust multi-robot formations [18]. Among the various control approaches investigated in the literature, coverage control stands out as a powerful methodology. It involves steering a team of robots to spread out across an environment in such a way that coverage is achieved in an optimal sense. This technique is closely related to the classical Lloyd's algorithm [19], and is tailored to accommodate the specific features and constraints of mobile sensor networks. The central idea of coverage control is to divide the environment into distinct regions, each assigned to a robot, and to optimize a cost function that quantifies how effectively the robot team can sense the environment. Solving this optimization problem yields robot configurations that maximize the collective sensing performance [1].

Additionally, the optimization framework can incorporate a density function to emphasize certain areas of interest within the environment. This function serves as a spatial weighting mechanism that guides the robots toward regions of higher importance. As a result, the final deployment configuration of the robots is influenced by the distribution of these areas, making the coverage strategy highly suitable for real-world scenarios such as surveillance or search and rescue, where focus on specific regions is crucial.

In this chapter, we introduce several control algorithms derived from the core principles of coverage-based control, adapted for deployment in realistic conditions, we describe a partitioning mechanism that enables each robot to independently compute its coverage region using only locally available sensing information, and, based on this decentralized partitioning, we present the coverage control method that empowers a group of robots

to explore and cover an unfamiliar environment, under limited sensing constraints. The methods and formulations introduced in this Chapter constitute the foundational building blocks for the remainder of this thesis. The decentralized coverage framework, performance metrics, and control abstractions presented here provide the common methodological backbone upon which the subsequent chapters build, enabling the systematic extension to time-varying objectives, constraint-aware execution, and non-planar coverage domains, as well as their validation through simulation and real-world experiments.

2.1 Notation and Definitions

This section introduces the notation, the concept of the Voronoi diagram, and the resolution mechanisms that will be employed throughout the remainder of this work.

2.1.1 Voronoi Diagrams

We denote by $\mathbb{N}$, $\mathbb{R}$, $\mathbb{R}_{\geq 0}$, and $\mathbb{R}_{>0}$ the set of natural, real, real non-negative, and real positive numbers. For $n \in \mathbb{N}$, we let $[n] = \{1, \ldots, n\}$. Given $x \in \mathbb{R}^n$, let $\|x\|$ be the Euclidean norm.

Let $\mathcal{G} = (\mathcal{U}, \mathcal{E})$ be a graph characterized by a set $\mathcal{U}$ of vertices and a set $\mathcal{E} \subseteq \mathcal{U} \times \mathcal{U}$ of edges. Given an edge $(i, j) \in \mathcal{E}$, then the vertex i is a neighbor of the vertex j. Let $\mathcal{N}_{\mathcal{G}}(i)$ be the set of neighbors of the vertex i in $\mathcal{G}$. A graph $\mathcal{G}$ is said to be *undirected* if $(i, j) \in \mathcal{E}$ implies $(j, i) \in \mathcal{E}$. Let $\mathbb{F}(\mathbb{R}^2)$ be the collection of finite point sets in $\mathbb{R}^2$. We can denote an element of $\mathbb{F}(\mathbb{R}^2)$ as $\mathcal{P} = \{p_1, \ldots, p_n\} \subset \mathbb{R}^2$, where $\{p_1, \ldots, p_n\}$ are points in $\mathbb{R}^2$. The *Voronoi partitioning* can be defined on a polygonal environment in $\mathbb{R}^2$, Figure 2.1 reports this nomenclature.

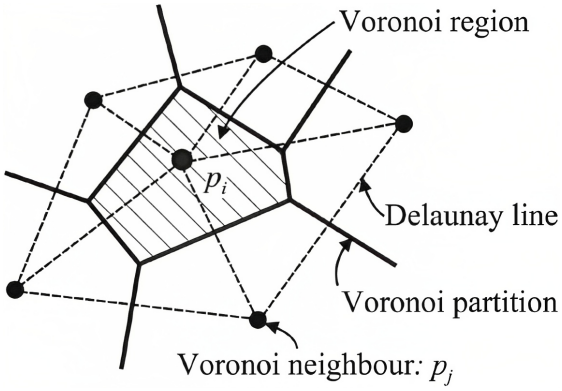


Figure 2.1: The Voronoi partitioning of a limited portion of environment around the entity i. The hatched area in the middle is the Voronoi cell V_i of the agent located in p_i.

In the rest of the chapter, we will use $Q \subset \mathbb{R}^2$ to denote such convex polygonal environment: it will be used, in particular, to denote the environment to be covered by the robots. An arbitrary point in Q is denoted by $q \in Q$.

Let then $\mathcal{P}$ be a set of n points $\{p_1, \ldots, p_n\}$ in Q. The *exact* Voronoi partitioning

generated by $\mathcal{P}$ consists in the set $\mathcal{V}(\mathcal{P}) = \{V_1(\mathcal{P}), \ldots, V_n(\mathcal{P})\}$, where:

$$V_i(\mathcal{P}) = \{q \in Q : \|q - p_i\| \leq \|q - p_j\|, \forall p_j \in \mathcal{P}\}. \tag{2.1}$$

In the following, for the sake of brevity, we will use the notation V_i to refer to $V_i(\mathcal{P})$. Two agents are said to be *Voronoi neighbors* if $V_i \cap V_j \neq \emptyset$. We refer to [20] for an extensive discussion on the Voronoi diagrams.

Let us define the *proximity graph* as a graph $\mathcal{G}$ in which the edge set $\mathcal{E}_\mathcal{G}$ depends on the location of the vertices. We denote by $\text{diag}(\mathcal{P} \times \mathcal{P})$ the diagonal of $\mathcal{P} \times \mathcal{P}$, namely the set of all pairs of the form (p, p) with $p \in \mathcal{P}$. Intuitively, this corresponds to all pairs in which the two entries coincide, and therefore represents the set of self-pairs associated with the points of $\mathcal{P}$. In this work we consider graphs defined by points $\{p_1, \ldots, p_n\}$ in $\mathbb{R}^2$. Hence, we can define a proximity graph function $\Gamma : \mathbb{F}(\mathbb{R}^2) \to \mathbb{G}(\mathbb{R}^2)$ that associates to $\mathcal{P} \in \mathbb{F}(\mathbb{R}^2)$ an undirected graph with $\mathcal{P}$ being the set of vertices and $\mathcal{E}_\Gamma(\mathcal{P})$ the set of edges, where $\mathcal{E}_\mathcal{G} : \mathbb{F}(\mathbb{R}^2) \to \mathbb{F}(\mathbb{R}^2 \times \mathbb{R}^2)$ has the property that $\mathcal{E}_\Gamma(\mathcal{P}) \subseteq \mathcal{P} \times \mathcal{P} \setminus \text{diag}(\mathcal{P} \times \mathcal{P})$. Given $p \in \mathbb{R}^2$ and $R \in \mathbb{R}_{>0}$, we denote the closed and open ball in $\mathbb{R}^2$ centered at p with radius R with $\overline{B}(p, R) = \left\{q \in \mathbb{R}^2 | \|q - p\| \leq R\right\}$ and $B(p, R) = \left\{q \in \mathbb{R}^2 | \|q - p\| < R\right\}$, respectively. Throughout the chapter, we will use 1_S to represent the indicator function, which is defined as $1_S(q) = 1$, if $q \in S$, and $1_S(q) = 0$, if $q \notin S$.

The following proximity graphs are relevant to our discussion:

1. the *R-disk graph* $\mathcal{P} \mapsto \mathcal{G}_{disk}(\mathcal{P}, R) = (\mathcal{P}, \mathcal{E}_{\mathcal{G}_{disk}}(\mathcal{P}, R))$, where the edges are defined as:
$$\mathcal{E}_{\mathcal{G}_{disk}}(\mathcal{P}, R) = \quad \{(p_i, p_j) \in \mathcal{P} \times \mathcal{P} \setminus \text{diag}(\mathcal{P} \times \mathcal{P}) \mid \|p_i - p_j\| \leq R\};$$

2. the *Delaunay graph* $\mathcal{P} \mapsto \mathcal{G}_D(\mathcal{P}) = (\mathcal{P}, \mathcal{E}_{\mathcal{G}_D}(\mathcal{P}))$, where the edges are defined as:
$$\mathcal{E}_{\mathcal{G}_D}(\mathcal{P}) = \quad \{(p_i, p_j) \in \mathcal{P} \times \mathcal{P} \setminus \text{diag}(\mathcal{P} \times \mathcal{P}) \mid V_i(\mathcal{P}) \cap V_j(\mathcal{P}) \neq \emptyset\};$$

3. the *R-limited Delaunay graph* (see Figure 2.2) $\mathcal{P} \mapsto \mathcal{G}_{LD}(\mathcal{P}, R) = (\mathcal{P}, \mathcal{E}_{\mathcal{G}_{LD}}(\mathcal{P}, R))$, where edges $(p_i, p_j) \in \mathcal{P} \times \mathcal{P} \setminus \text{diag}(\mathcal{P} \times \mathcal{P})$ are defined as:
$$(V_i(\mathcal{P}) \cap \overline{B}(p_i, R/2)) \cap (V_j(\mathcal{P}) \cap \overline{B}(p_j, R/2)) \neq \emptyset.$$

The proximity graphs considered in this Chapter encode local interactions among points based on either metric distance, geometric adjacency, or a combination of both. The *R-disk graph* models range-limited interactions by connecting points within a fixed Euclidean distance, the *Delaunay graph* captures neighborhood relations induced by the Voronoi tessellation of the point set, and the *R-limited Delaunay graph* restricts such geometric adjacency to bounded regions, enforcing locality while preserving the structural properties of the Delaunay graph. For each of the proximity graphs $\mathcal{G}_{disk}$, $\mathcal{G}_D$, and $\mathcal{G}_{LD}$, we can define the respective set of neighbors of the point p as $\mathcal{N}_\mathcal{G}(p, \mathcal{P}) = \{q \in \mathcal{P} \mid (p, q) \in \mathcal{E}_\mathcal{G}(\mathcal{P} \cup \{p\})\}$.

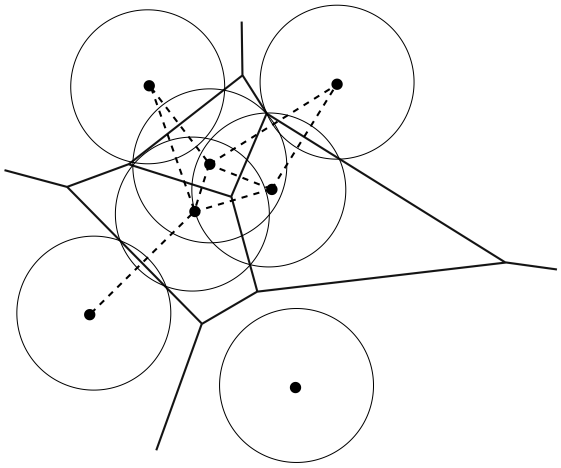


Figure 2.2: The Voronoi partitioning and the R-limited Delaunay graph (dashed line) with seven seeds. The circles represents the balls with radius $R/2$ around every seed.

2.1.2 Fortune's Algorithm

The construction of Voronoi diagrams, as introduced in the previous section, plays a fundamental role in computational geometry and distributed robotic systems. Among the various algorithms available for their construction, *Fortune's algorithm* is one of the most efficient and widely used methods for computing Voronoi diagrams in $\mathbb{R}^2$ [21]; this is a sweep-plane algorithm introduced by Steven Fortune in 1987. It computes the exact Voronoi diagram of a set $\mathcal{P} = p_1, \ldots, p_n \subset \mathbb{R}^2$ of n distinct sites in $\mathcal{O}(n \log n)$ time using a vertical sweep line that moves from top to bottom of the plane. The main idea is to incrementally build the Voronoi diagram by maintaining a dynamic data structure known as the *beach line* and a priority queue of future events.

The algorithm relies on two key event types:

- **Site events**, which occur when the sweep line reaches a new point $p_i \in \mathcal{P}$. These events cause new arcs to be added to the beach line.
- **Circle events**, which occur when a point on the beach line disappears, corresponding to the creation of a Voronoi vertex. These are detected when three consecutive arcs on the beach line become co-circular.

The *beach line* is a piecewise parabolic curve that separates the processed (above the sweep line) and unprocessed (below the sweep line) portions of the diagram. Each arc on the beach line corresponds to a portion of a Voronoi cell boundary under construction. When a site event is processed, the algorithm inserts a new arc into the beach line, and when a circle event is triggered, it removes the appropriate arc and records the Voronoi vertex.

The algorithm maintains:

- A *priority queue* $\mathcal{Q}$, containing all future site and circle events, ordered by their y-coordinate.
- A *balanced binary search tree* $\mathcal{T}$, representing the beach line structure.

Each edge of the resulting Voronoi diagram is traced during the sweep and stored as a sequence of line segments defined by Voronoi vertices.

The main advantages of Fortune's algorithm are:

1. **Optimal complexity**: The algorithm achieves the lower bound $\mathcal{O}(n \log n)$ for planar subdivision problems.

2. **Robustness**: It handles general inputs and avoids unnecessary computations by maintaining only the relevant portions of the diagram during construction.

3. **Incrementality**: While the sweep line is not incremental in the sense of dynamic insertion of new sites after the sweep, it incrementally constructs the diagram during the sweep process.

Despite its conceptual complexity, the algorithm is well-suited for implementation in computational geometry libraries and is commonly used in practice.

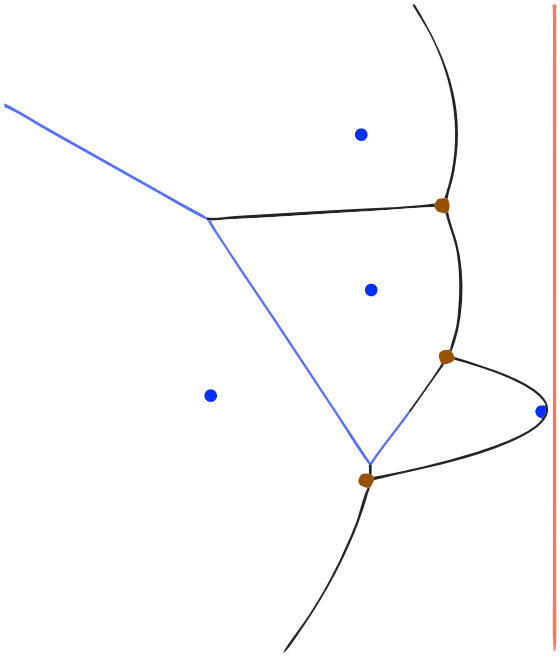


Figure 2.3: Illustration of Fortune's sweep line algorithm. The current sweep line is shown in black, while the beach line is shown in red. The blue point denotes the event currently being processed, and the blue segments represent the portion of the Voronoi diagram that has already been constructed. The beach line separates the processed region above the sweep line from the unprocessed region below it.

Formally, let us define the Voronoi diagram $\mathcal{V}(\mathcal{P})$ of a point set $\mathcal{P}$ as in Equation (2.1). Fortune's algorithm constructs the full set $\mathcal{V}(\mathcal{P}) = V_1, \ldots, V_n$ by processing all events in $\mathcal{Q}$ until completion.

The output of the algorithm is a set of Voronoi edges, vertices, and cells, which can be post-processed to extract adjacency information (e.g., Delaunay edges) and graph structures like $\mathcal{G}_D(\mathcal{P})$. The dual graph of the Voronoi diagram, i.e., the Delaunay triangulation, emerges naturally as a byproduct of the algorithm.

For further details and formal analysis, we refer to [20, 21].

2.2 Coverage Control

2.2.1 Vehicle Model

We study a multi-robot system composed of n agents operating in a two-dimensional space, we consider the z negligible for the behaviour in this phase. Each robot is modeled as a single integrator, with its position denoted by $p_i \in \mathbb{R}^2$, and its dynamics governed by:

$$\dot{p}_i = u_i, \tag{2.2}$$

where $u_i \in \mathbb{R}^2$ represents the control input for robot i, for all $i = 1, \ldots, n$. The collective configuration of the robots is expressed as the set $\mathcal{P} = p_1, \ldots, p_n$. It is worth noting that, although the single integrator is a simplified abstraction, it remains highly effective for real-world robotic control. When coupled with appropriate trajectory tracking strategies, this model can be used to generate velocity commands for a variety of robotic platforms, including wheeled mobile robots [22] and unmanned aerial vehicles [23].

2.2.2 Lloyd's Algorithm in Multi-Agent Cases

In the coverage control problem a performance function is defined to be maximized in order to obtain the optimal coverage of the group of robots over Q. The performance function is chosen to model how reliable is the measurement, at point $q \in Q$, performed by robot i whose position is p_i, as a function of the distance $\|q - p_i\|$. Therefore, the performance function can be defined as a non-increasing differentiable function $f(\|q - p_i\|) : \mathbb{R}_{\geq 0} \to \mathbb{R}$. Moreover, an integrable probability density function $\phi : Q \to \mathbb{R}_{\geq 0}$ can be defined in order to characterize the portions in Q where an event of interest occurs. Namely, ϕ is used to capture the relative importance of determined areas in the environment. For example, $\phi(q_1) > \phi(q_2)$ implies that q_1 is more important than q_2, or that the area around q_1 has a higher probability of being an area of interest.

The following *optimization* function $H : Q^n \to \mathbb{R}$ can then be formulated:

$$\mathcal{H}(\mathcal{P}) = \int_Q \max_{i \in \{1,\ldots,n\}} f(\|q - p_i\|)\phi(q)dq, \tag{2.3}$$

with a better coverage corresponding to a higher value of the function–which strictly depends on *R-limited Delaunay graph* formulation and size of the multi-agent system. In order to solve the maximization problem, the environment is assumed to be partitioned into regions, where each node is in charge of covering its own region: the notion of Voronoi tessellation, introduced in (2.1) in Section 2.1, becomes then fundamental. Thus, given an *exact* Voronoi partitioning $\mathcal{V}(P)$ of the environment Q in, so called, n Voronoi cells $\{V_1, \ldots, V_n\}$, the *optimization* function $H : Q \to \mathbb{R}$ can be formulated as follows:

$$\mathcal{H}(\mathcal{P}, \mathcal{V}) = \sum_{i=1}^{n} \int_{V_i} f(\|q - p_i\|)\phi(q)dq, \tag{2.4}$$

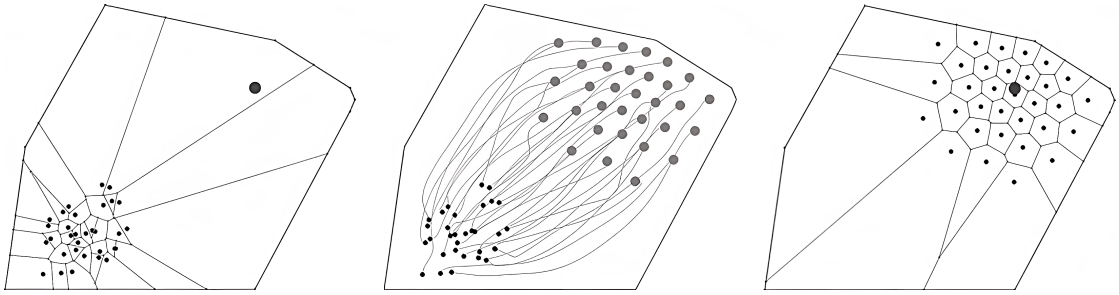


Figure 2.4: This figure is summarizing the coverage control methodology from Cortes *et al.* [1]. The control strategy is applied to 32 agents deployed in a convex polygonal environment, with a Gaussian density distribution centered in the top right area of the environment. The left figure illustrates the initial robots locations and the respective Voronoi diagrams. The middle figure illustrates the robots trajectories. The right figure illustrates the final robots locations and the respective Voronoi diagrams. To be noted that this case is constrained under Gaussian domain definitions, that are defined later in the chapter.

where each node is in charge of covering its own cell, and with a better coverage corresponding to a higher value of the function. In the literature, the performance function is usually chosen as: $f(x) = -x^2$ [1, 24, 25]. Therefore, the optimization function becomes:

$$\mathcal{H}_{\mathcal{V}}(\mathcal{P}) = -\sum_{i=1}^{n} \int_{V_i} \|q - p_i\|^2 \phi(q) dq = -\sum_{i=1}^{n} J_{V_i,p_i}, \tag{2.5}$$

where $\mathcal{H}_{\mathcal{V}}(\mathcal{P}) = \mathcal{H}(\mathcal{P}, \mathcal{V})$ and J_{V_i,p_i} indicates the polar moment of inertia of the region V_i about the point p_i. In order to solve the optimization problem, the gradient of the optimization function can be computed, obtaining:

$$\frac{\partial \mathcal{H}_{\mathcal{V}}}{\partial p_i}(\mathcal{P}) = 2M_{V_i}(C_{V_i} - p_i), \tag{2.6}$$

where M_{V_i} and C_{V_i} denote respectively the mass and the center of mass with respect to the density function ϕ of the Voronoi cell $V_i \subset Q$ of the robot i in position p_i. Therefore, the mass and centroid can be computed as follows:

$$M_{V_i} = \int_{V_i} \phi(q) dq, \; C_{V_i} = \frac{1}{M_{V_i}} \int_{V_i} q\phi(q) dq. \tag{2.7}$$

For more details we refer the reader to [1, 25].

According to (2.6), the configuration of points $\mathcal{P}$ which maximizes the optimization function $\mathcal{H}_{\mathcal{V}}(\mathcal{P})$ coincides with the *centroids* of the respective Voronoi cells. In other words, the solution to the maximization problem is achieved when each agent is located at the centroid of its Voronoi region, such that $p_i = C_{V_i}, \; \forall i$. Such configurations are called *centroidal Voronoi configurations*, see [26]. $\mathcal{H}_{\mathcal{V}}(\mathcal{P})$ can be used as a Lyapunov-like function and we can design a control law that drives the group of robots to a centroidal Voronoi configuration, locally maximizing the coverage in the environment. In particular, the control input for each robot can be computed according to the Lloyd algorithm [1], as follows:

$$u_i = k(C_{V_i} - p_i), \tag{2.8}$$

where $k \in \mathbb{R}_{>0}$ is a proportional gain.

The exact Voronoi partitioning $\mathcal{V}(\mathcal{P})$ of the region Q is assumed to be continuously updated. Thus, it is is considered that the region Q is known or that each robot is able to measure it. The network of the group of robots is defined by the Delaunay proximity graph $\mathcal{G}_D(\mathcal{P})$. To compute the i-th component of the $\mathcal{H}_{\mathcal{V}}(\mathcal{P})$ function, it is necessary for the robot i to know its own position and the positions of its neighbors over the graph $\mathcal{G}_D(\mathcal{P})$. Usually the positions of all the robots are assumed to be known or measurable. A modified optimization function is then introduced in [24], where coverage is achieved considering the R-limited Delaunay graph $\mathcal{G}_{LD}(\mathcal{P}, R)$, where two robots are neighbors if their respective Voronoi cells are neighboring, *and* if their distance is smaller than the communication range R. It is worth noting, however, that this graph is built upon the exact Voronoi partitioning $\mathcal{V}(\mathcal{P})$, whose computation requires, as already discussed, full knowledge of the region Q and at least of the Delaunay neighbors. In Chapter 3 we will show how an alternative partitioning approach overcomes these issues, requiring only local knowledge of the portion of the region Q and neighbors that can be directly observed by each robot. Centroidal Voronoi Tessellation (CVT) provide the theoretical foundation for coverage optimization and were formally introduced in [27].

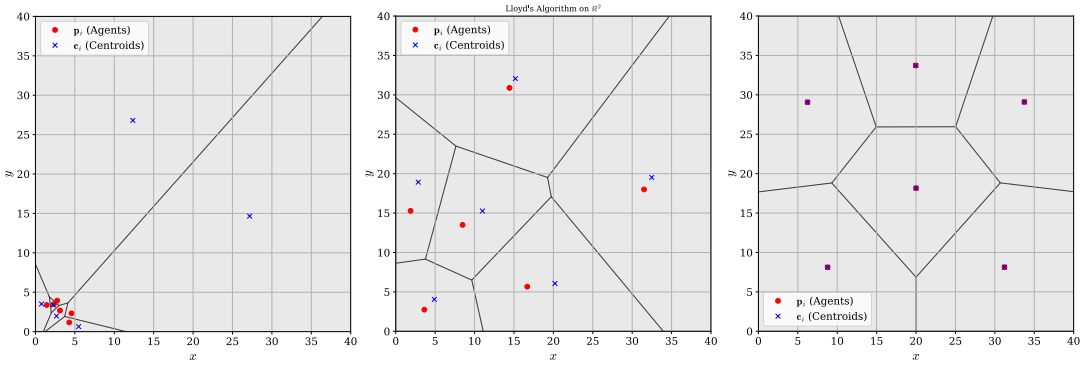


Figure 2.5: Lloyd's Algorithm applied to a configuration of 6 agents in a convex polygonal environment. The figure on the left shows the initial positions of the drones and the result of the first centroid computation. The central figure illustrates the configuration after several iterations, while the figure on the right shows the final configuration at convergence, corresponding to a CVT condition.

Lloyd's algorithm thus provides a simple and intuitive iterative procedure to drive a multi-agent system toward a centroidal Voronoi configuration by alternately computing Voronoi partitions and moving agents toward their corresponding centroids. Its appeal lies in its fully distributed structure, geometric interpretability, and provable convergence properties under idealized assumptions, such as exact knowledge of the environment, continuous partition updates, and unconstrained agent motion [1, 26]. However, these assumptions are rarely satisfied in practical multi-UAV deployments, where sensing is limited, communication is range-constrained, and vehicle dynamics impose additional constraints on the execution of the control law. As a result, while Lloyd's algorithm constitutes the theoretical backbone of coverage control, significant adaptations are required to enable its reliable deployment in realistic scenarios. Addressing these limitations and extending Lloyd-based coverage control toward sensing-limited, constraint-aware, and dynamically evolving environments is the focus of the subsequent chapters. Figure 2.5 shows the iteration of the Lloyd's algorithm.

2.2.3 Lloyd's Algorithm under Gaussian Density Functions

The formulation introduced so far assumes a uniform importance distribution over the space, corresponding to $\phi(q) = 1$. In this section, we analyze the performance of the coverage control algorithm in the presence of non-uniform importance distributions, where specific points of interest are assigned higher weights and therefore exert a stronger influence on the resulting coverage configuration.

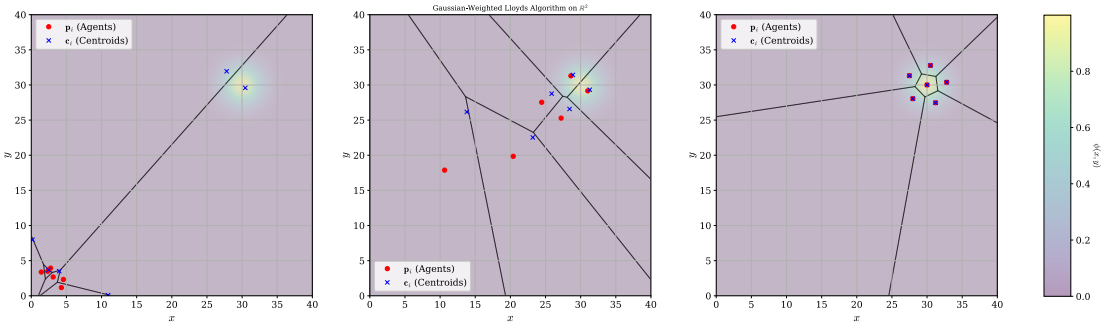


Figure 2.6: Lloyd's Algorithm applied to a configuration of 6 drones in a convex polygonal environment, with arbitrary $\phi(q)$.

In the proposed control strategy we consider a probability density function $\phi : Q \to \mathbb{R}_{>0}$ in order to measure the position in Q where an event of interest occurs. Such probability density function may be known a priori to all the robots, or may be estimated, at runtime, based on local measurements, as discussed in [25]: hence, without loss of generality, we will hereafter assume the probability density function to be known to all the robots.

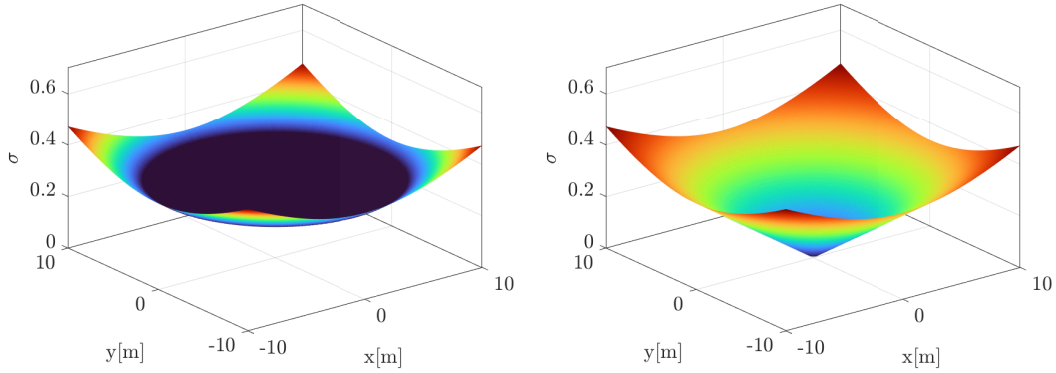
According to the control law defined in (2.8), each robot is required to evaluate the weighted mass and centroid of its assigned region, as determined by the Voronoi partition. In this work, the density function ϕ is assumed to follow a Gaussian distribution[1]. Gaussian densities exhibit several properties that significantly simplify the analytical computation of both mass and centroid. For instance, the approach presented in [28] exploits a line–integration technique for Voronoi cells, reducing the computational burden typically associated with spatial discretization methods.

Given $\mu \in Q$ mean of the Gaussian Distribution, σ its standard deviation, the formulation of the probability density function is the canonical:

$$\phi(x) = exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{2.9}$$

It is worth emphasizing that the introduction of a Gaussian density function does not alter the structure of Lloyd's algorithm itself. The iterative procedure remains unchanged, consisting of alternating Voronoi partition computation and motion toward the corresponding centroids. The only modification lies in the definition of the centroidal quantities, which are now computed with respect to the weighted density function ϕ. As a result, Lloyd's algorithm naturally extends to the non-uniform case by driving the

[1]The proposed formulation readily generalizes to arbitrary probability density functions through mixtures of Gaussian components.



(a) $\sigma = 0.3$ **(b)** $\sigma = 0.01$

Figure 2.7: Examples of two surfaces calculated with (2.9).

agents toward a centroidal Voronoi configuration weighted by the Gaussian importance distribution, yielding a coverage arrangement biased toward regions of higher relevance. Figure 2.6 shows the iteration of Lloyd's algorithm in the presence of a Gaussian Probability Density Function (PDF), while Figure 2.7 the difference of distributions at varying of σ.

2.3 Unmanned Aerial Vehicles: Quadrotor Control

In Section 2.2.1, a simple vehicle model was introduced for the sake of developing the coverage control algorithm. In particular, the coverage control procedures described in Section 2.2 rely on a single-integrator model of the form $\dot{p}_i = u_i$ to steer each agent toward the centroid of its local region. While this abstraction is appropriate for algorithmic design, real quadrotors are underactuated rigid–body systems evolving on the nonlinear configuration manifold SE(3)—i.e. Special Euclidean Group in 3D. To implement the coverage control laws on physical aerial robots, a Low Level Controller (LLC) is required to track desired trajectories or velocity commands while ensuring stable and robust flight dynamics. In this section we summarize the quadrotor dynamic model and the geometric tracking controller that constitute the foundation of the control architecture adopted in all our experiments. The material follows the formulation proposed in [29], which provides a globally defined representation on SE(3) together with almost–global stability guarantees.

2.3.1 Quadrotor Dynamics on SE(3)

A quadrotor is defined as a vehicle consisting of four fixed-pitch rotors arranged at the vertices of a square frame, producing thrust and reaction torques for control of translational and rotational motion. Let $\{\mathbf{i}_1, \mathbf{i}_2, \mathbf{i}_3\}$ denote the inertial frame and $\{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\}$ the body-fixed frame whose origin is located at the center of mass. The configuration of the vehicle is given by its position $x \in \mathbb{R}^3$ and attitude $R \in \mathrm{SO}(3)$—i.e. Special Orthogonal Group in 3D. We denote by $v \in \mathbb{R}^3$ the inertial-frame velocity of the center of mass and by $\Omega \in \mathbb{R}^3$ the angular velocity expressed in the body frame. A schematic

representation of the model is shown in Figure 2.8 taken from [29].

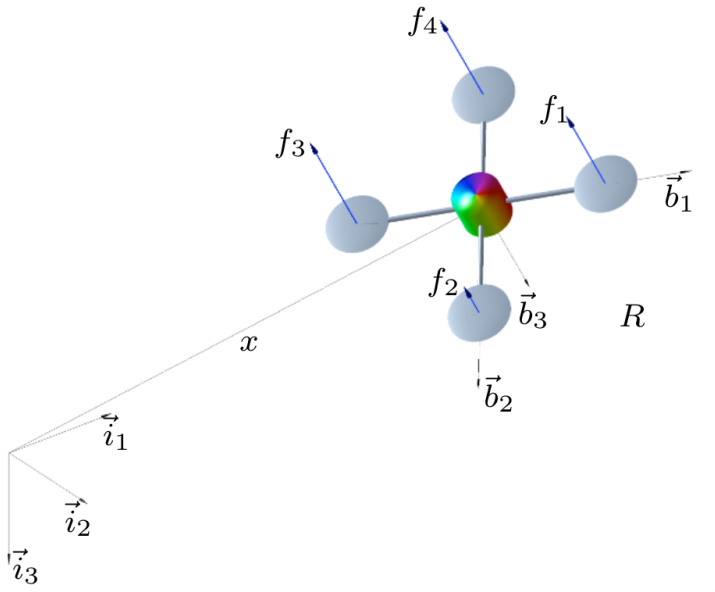


Figure 2.8: Schematic of a quadrotor with body-fixed axes and thrust directions from Lee *et al.*, this model is adopted in this whole work.

Let m denote the mass, $J \in \mathbb{R}^{3\times3}$ the inertia matrix, and d the length from the center of mass to each rotor. The thrust produced by rotor i is f_i, acting along the negative $\mathbf{b}_3$ axis. Define the total thrust:

$$f = \sum_{i=1}^{4} f_i, \tag{2.10}$$

and the total moment $M \in \mathbb{R}^3$, obtained through the linear allocation map. Let $M = [M_1\ M_2\ M_3]^T \in \mathbb{R}^3$ denote the total control moment expressed in the body-fixed frame, where M_1, M_2, and M_3 are the moments about the b_1, b_2, and b_3 axes, respectively. Moreover, let d be the distance from the center of mass to each rotor, and let $c_{\tau f}$ denote the proportional coefficient relating rotor thrust to the corresponding reaction torque.

$$\begin{bmatrix} f \\ M_1 \\ M_2 \\ M_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -d & 0 & d \\ d & 0 & -d & 0 \\ -c_{\tau f} & c_{\tau f} & -c_{\tau f} & c_{\tau f} \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix}, \tag{2.11}$$

which is invertible for $d \neq 0$ and $c_{\tau f} \neq 0$ [29]. The equations of motion of the quadrotor evolve on SE(3):

$$\dot{x} = v, \tag{2.12}$$

$$m\dot{v} = mge_3 - fRe_3, \tag{2.13}$$

$$\dot{R} = R\hat{\Omega}, \tag{2.14}$$

$$J\dot{\Omega} + \Omega \times J\Omega = M, \tag{2.15}$$

where $\hat{\Omega}$ denotes the hat map sending vectors in $\mathbb{R}^3$ to the Lie algebra SO(3). These equations capture the essential underactuated structure of the quadrotor, while four inputs (f, M) are available, the system has six degrees of freedom. Consequently, trajectory tracking requires a controller that coordinates thrust and attitude in a nonlinear manner.

2.3.2 Geometric Tracking Control on SE(3)

Geometric control provides a coordinate-free framework for stabilizing systems that evolve on nonlinear manifolds such as SO(3) and SE(3) [30]. The controller proposed in [29] avoids the singularities associated with Euler angles and the double-cover ambiguity of quaternions, which are known to lead to topological obstructions and unwinding phenomena in attitude control [31], and it provides almost–global stability guarantees on attitude.

The goal is to track a desired trajectory $x_d(t)$ and a desired heading direction $b_{1d}(t)$. The translational subsystem is stabilized by choosing a desired direction for the thrust vector:

$$b_{3d} = \frac{-k_x e_x - k_v e_v - mge_3 + m\ddot{x}_d}{\|-k_x e_x - k_v e_v - mge_3 + m\ddot{x}_d\|}, \tag{2.16}$$

where $e_x = x - x_d$ and $e_v = v - \dot{x}_d$. The attitude command is constructed by imposing that the vehicle's body axis $\mathbf{b}_3$ aligns with b_{3d} while $\mathbf{b}_1$ tracks the heading direction:

$$R_d = [\, b_{2d} \times b_{3d},\ b_{2d},\ b_{3d}\,], \qquad b_{2d} = \frac{b_{3d} \times b_{1d}}{\|b_{3d} \times b_{1d}\|}. \tag{2.17}$$

Attitude and angular velocity tracking errors are defined as [29]:

$$e_R = \frac{1}{2}(R_d^T R - R^T R_d)^\vee, \tag{2.18}$$

$$e_\Omega = \Omega - R^T R_d \Omega_d. \tag{2.19}$$

The control inputs are then given by:

$$f = -\left(-k_x e_x - k_v e_v - mge_3 + m\ddot{x}_d\right) \cdot (Re_3), \tag{2.20}$$

$$M = -k_R e_R - k_\Omega e_\Omega + \Omega \times J\Omega - J(\hat{\Omega} R^T R_d \Omega_d - R^T R_d \dot{\Omega}_d). \tag{2.21}$$

A block diagram of the controller is shown in Figure 2.9.

2.3.3 Stability Properties

A fundamental advantage of the geometric controller is the almost–global nature of its stability guarantees. Lee *et al.* show that the attitude subsystem is exponentially stable

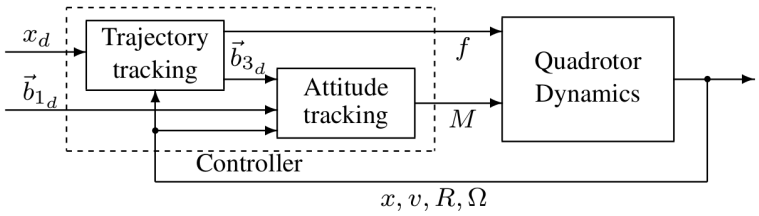


Figure 2.9: High-level structure of the geometric tracking controller on SE(3).

for any initial attitude error less than 180°, and the full translational–rotational dynamics are exponentially stable for initial attitude errors below 90° [29]. Furthermore, for any initial condition satisfying:

$$1 \leq \Psi(R(0), R_d(0)) < 2, \tag{2.22}$$

the tracking errors converge exponentially after a finite transient, demonstrating *almost–global exponential attractiveness*. These properties make the controller especially suitable for real-world operations, where large disturbances or aggressive maneuvers may occur. In contrast to earlier attitude stabilization strategies for quadrotors based on more traditional representations and control techniques [32], the SE(3) formulation ensures well-posed behavior over almost the entire configuration space.

2.3.4 Integration With Coverage Control

The geometric controller provides the essential link between the idealized single-integrator control law used in coverage algorithms and the physical dynamics of quadrotors. Coverage control formulations typically assume simplified kinematic models, which are convenient for analysis and decentralized coordination but do not capture the nonlinear and underactuated nature of real aerial vehicles. The geometric controller bridges this gap by enabling the execution of coverage-generated motion references on physical platforms.

Given a desired velocity or position reference generated by Lloyd's algorithm, the SE(3) controller computes a feasible thrust and attitude command to be tracked by the onboard autopilot. This mapping ensures that the commanded motion is dynamically admissible and compatible with the actuation limits of the quadrotor. In the implemented architecture, the nominal geometric tracking law is augmented with an integral compensation term to mitigate the effect of constant or slowly varying disturbances, such as aerodynamic biases, model mismatch, or persistent external perturbations. More specifically, the translational tracking term can be written as $-k_x e_x - k_v e_v - k_I e_I - mge_3 + m\ddot{x}_d$, where e_I denotes the integral state associated with the translational tracking error. The additional integral action reduces steady-state tracking errors and improves the practical robustness of the controller when executing coverage-induced references on real aerial platforms. As a result, the coverage layer can still be designed independently from the low-level flight dynamics, while the augmented tracking controller improves the fidelity with which the high-level coverage commands are reproduced in practice. This is par-

ticularly relevant in outdoor experiments, where non-ideal effects may otherwise degrade the execution of the decentralized coverage behavior.

This abstraction improves modularity and portability across different aerial platforms. In the field experiments that will be presented in Chapter 3, this controller was employed to realize the decentralized coverage behavior on a fleet of quadrotors operating in an outdoor environment, demonstrating the effectiveness of the proposed integration in realistic conditions. The Chapters in which the experimental results are presented also illustrate how the transformation between the SE(3) controller output and the motor RPM commands is implemented in practice.

Chapter 3

Coverage Control for Large-Scale Teams of Aerial Drones

Coverage control constitutes a fundamental primitive in multi-robot aerial systems, enabling the execution of a wide range of higher-level missions such as surveillance, monitoring, and exploration. While extensive literature exists on optimal and centralized coverage strategies, the deployment of large-scale aerial teams in real-world environments raises critical challenges related to scalability, communication constraints, and robustness. As the number of agents increases, approaches relying on global information and centralized computation often become impractical, leading to reduced efficiency, increased latency, and limited reactivity to local disturbances.

To this end, in this Chapter we analyze the control methodology proposed in Section 2.2 for area coverage that is fully distributed and able to mitigate the above factors while still providing high performance. We focus on fully distributed coverage control strategies tailored to large-scale teams of aerial drones operating under limited sensing and communication capabilities. Building upon the theoretical foundations introduced in Chapter 2 with Voronoi-based approach, we investigate coverage methodologies that rely exclusively on local interactions, thereby preserving scalability while maintaining satisfactory coverage performance.

To validate the proposed framework, we conduct an extensive ablation study over a large set of simulations, evaluated using a set of dedicated performance metrics. In addition, the algorithm is validated through real-world experiments with quadrotor platforms, including proof-of-concept field deployments involving up to seven aerial units. These experimental results demonstrate the practical feasibility of deploying decentralized coverage controllers on large aerial teams operating in unstructured outdoor environments.

3.1 Literature Review

The literature is flourishing with strategies proposing multiple implementation of coverage control performed by a group of mobile robots—either ground, aerial, marine or

hybrid. Recent work has also explored fast decentralized exploration strategies for large multi-UAV teams operating in complex natural environments [33]. Amongst these, Cortes *et al.* [1] proposes decentralized coordination algorithms for groups of mobile agents based on Voronoi diagrams, as well as [26], [34], [35]; Bullo *et al.* [36] and Laventall *et al.* [37] propose strategies based on proximity graphs, while Zhang *et al.* investigate optimal placement strategies [38]. Alternative decentralized exploration strategies based on random walks have been proposed for swarm mapping and area discovery tasks [39].

These control strategies provide a methodology to guarantee convergence of the positions of the robots in the team to a configuration that maximizes the area covered in the environment. The main drawback is related to the fact that the computation of each robot's control input requires the computation of the Voronoi partitioning of the entire environment, which requires global knowledge and/or global communication and information exchange [40], [25]. However, in many practical application scenarios, robots may not have access to global information or to a reliable all-to-all communication network: therefore, methods were proposed to define the Voronoi partitioning in a range-constrained scenario. The distributed methodology introduced in [1, 24] calculate the Voronoi partition over a limited sensing network, exploiting the algorithm presented in [41], which consists in gradually refining the globally computed partition by utilizing an adjustable sensing range and incrementally increasing the sensing range. However, in practical applications, limited sensing capabilities are typically a hard constraint, and rarely the sensing radius can be adjusted to detect sufficient information to allow computation of the Voronoi partition.

To overcome these issues, the methodology proposed in section 2.2 aims to implement coverage control in a totally distributed manner, letting each robot compute its own control input based on locally available information only. In particular, the control strategy allows a team of mobile robots, moving in a two-dimensional environment, to optimally cover a bounded region with unknown feature. Robots are considered with limited sensing capabilities, and without access to reliable high-bandwidth communication infrastructures.

The aim of this work is to experimentally evaluate the performance of the coverage control architecture discussed in 2.2 deployed on a large team of drones for outdoor operations. In this section, we perform a deep and extensive analysis of both realistic simulations and real field tests with a fleet of quad-rotor drones in an unstructured environment as the Abu Dhabi desert. Thanks to these, we assess the scalability of the control methodology and its performance in the presence of real-world constraints and limitations such as localization error coming from noisy Global Positioning System (GPS) readings. In order to guarantee the statistical relevance of the simulated results, we perform an ablation study of the main parameters with randomized initialization conditions repeated over multiple runs. We note that this represents the first large-scale deployment of a team of drones performing fully decentralized and distributed coverage control in an outdoor environment.

3.2 Tests and Setups

The aim of the experiments is to evaluate the performance of a team of drones controlled by a decentralized, limited-range, coverage control strategy (Section 2.2), deployed in a large outdoor environment. For this purpose, field trials and high-fidelity dynamic simulations were set up, to achieve a statistically relevant characterization. Details of the methods are provided in the following paragraphs.

3.2.1 Simulation Protocol

Simulated experiments have been performed using the Flightmare Quadrotor Simulator [42], a photo-realistic simulator, with integrated physics, developed by the Robotics and Perception group at UZH. The control algorithm described in Section 2.2 has been implemented using the Robot Operating System (ROS) [43] on separate threads, one for each drone. Each robot thread also handles the LLC for the quad-rotor fly, the initial take-off, and the sharing of information. A python script acting as a parallel daemon manages the execution of the simulation batches and collects the data needed to assess performances. The simulations have been performed on an Intel Xeon Platinum 8280M CPU@2.70 GHz with 4 processors, for a total of 112 cores and 512 gigabytes of RAM. Each simulation collects the global position for each drone over a time interval of 120 seconds—empirically set at the end of the simulation. To guarantee the independence of the proposed methodology from the initial conditions, drones are spawned inside the bounded environment in randomized positions: for every simulation run, the vehicles are spawned within the simulation environment in a neighborhood of the center of the coverage area with a random displacement. The final position is bounded to be at least 0.5m from any of the already positioned vehicles. After arming and take-off of all the vehicles at the same altitude, the coverage control algorithm takes over and steers the drones toward their wanted position thanks to the velocities by Equation (2.8). The simulator makes use of the RPG Quadrotor control [44], which allows for generating desired orientation, collective thrust command, body rates, angular accelerations, and angular jerk to generate the control values of the four motors from the requested velocity command.

The measurements of the positions of the other vehicles, used to feed the control function, are added to a noise value generated by a normal distribution $\mathcal{N}(0, 1)$ to simulate realistic measurement error. Ten simulations have been performed by combining different parameters expressed by a tuple (n, a, r) respectively representing: the number of drones $n \in N = \{8, 32, 64\}$; the size of the operational area $a \in A = \{50m, 100m, 200m\}$; and the sensitivity range $r \in R = \{10m, 50m, 100m, 500m\}$.

3.2.2 Field Test Protocol

Real-world experiments have been performed in a remote location in the Al Khaznah region of the Abu Dhabi emirate in the UAE. The platform selected as the flying unit

is depicted in Figure 3.1. It is a custom-built quad-rotor developed by the company Fly4Future, consisting of a standard DJI frame of 450mm, a Pixhawk4 autopilot with custom firmware, and an Intel NUC with i7 processor and 16 gigabytes of RAM as an on-board computer. The latter is used to run the high-level software control stack that, in this particular case, runs on the MRS UAV system openly available for research purposes and released by the Multi-robot Systems Group at the Czech Technical University in Prague [45].

Figure 3.1: Figure of the MRS F450, the quad-rotor platform selected for the experiments

The robotic unit is equipped with several sensors among which, for the specific use case we analyze here, we highlight a Global Positioning System module for self-localization—as the one coming with the Pixhawk autopilot bundle—and a XBee devices for communication among the units and information sharing. The latter, makes use of an ad-hoc peer-to-peer proprietary mesh-network called DiGiMesh that we limit to be single-hop to force the range communication constraint. We employed omni-directional low-power antennas on board all drones, each of which runs on 900Mhz empirically estimated to guarantee a good connectivity among all the units in the group. The vehicle controller [29] allows to generate the desired orientation, collective thrust command, body rates, angular accelerations, and angular jerk based on the coverage control, as mentioned in Section 2.3. The experiments were carried out using 7 vehicles, in an area of 50×50 m; as previously described, the communication devices used offer full connectivity but, to evaluate the behavior of the control strategy we manually limit the effect of the sensing range r with the same values used for the simulations— i.e., $R \in \{10m, 50m, 100m\}$. Three experiment runs have been performed for each value of the sensing range. As in the simulated scenario, the control algorithm proposed in Section 2.2 runs in a distributed way on board of each drone. The position information of neighboring units needed by the coverage control are solely provided through radio communication and only by the aerial units—i.e., no external or central computer is used. Figure 3.2 shows the trajectories of

the UAVs in one of the experiments carried out; following the takeoff procedure to bring the vehicles to the same altitude, the coverage control function is performed to determine the new positions of the vehicles.

The speed control of Equation 2.2 is limited to $2m/s$ for safety reasons, taking into account the number of vehicles involved in the experiment and the degree of freedom that the drones have to move in various directions.

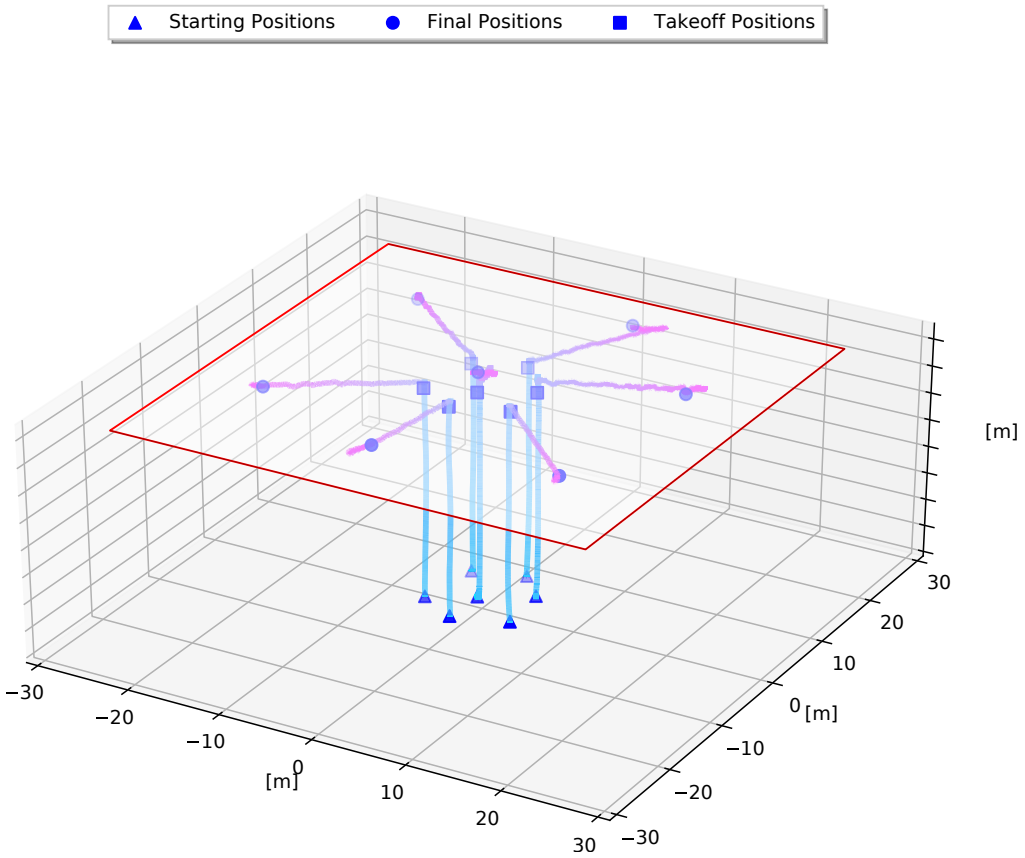


Figure 3.2: Vehicles trajectories of a real experiment with 7 drones

3.3 Metrics

In order to evaluate the overall performance of the system, the same metrics were collected both in simulations and in field tests. In particular, we consider the following three main aspects: *(i)* optimality of the final configuration, *(ii)* efficiency in the coverage of the environment, and *(iii)* effectiveness in the use of the robots.

3.3.1 Optimality of The Configuration

As is typically done in the literature, the performance of the coverage control algorithm is evaluated computing the value of an optimization function $\mathcal{H}_{\mathcal{V}}(\mathcal{P})$ as introduced in (2.5), whose value indicates how close the displacement of the robots is to the optimal configuration, that is the centroidal Voronoi configuration. We report the optimization function

in (3.1), referring to it as $\mathcal{H}(\mathcal{P})$ simplifying the notation.

Considering the positions $\mathcal{P}$ of the robots, it is possible to partition the environment Q into n *range-unlimited* Voronoi cells $\{\bar{V}_1, \ldots, \bar{V}_n\}$, where the i-th cell is computed using position p_i as a seed. The optimization function is then defined as follows:

$$\mathcal{H}(\mathcal{P}) = -\sum_{i=1}^{n} \int_{\bar{V}_i} \|q - p_i\|^2 \phi(q) dq \tag{3.1}$$

As discussed in [1, 25], such definition allows to quantify how close the configuration of the robots is to the centroidal Voronoi configuration, defined considering, for each point $q \in Q$, the value of the PDF $\phi(q)$. Since $\mathcal{H}(\mathcal{P})$ is defined as the negative of the weighted quadratic coverage cost, its values are non-positive, and better coverage configurations correspond to larger values of $\mathcal{H}(\mathcal{P})$, i.e., values closer to zero. The maximum of $\mathcal{H}(\mathcal{P})$ is attained at a centroidal Voronoi configuration, although its numerical value depends on the specific domain, density function, and number of robots.

3.3.2 Coverage Effectiveness

Considering the limited-range sensing capabilities of the robots, in order to assess how well the environment Q is covered by the set of individual robot non-overlapping limited-range cells $\{V_1, \ldots, V_n\}$, we propose to evaluate the ratio $\mathcal{A}(\mathcal{P})$ of the integral of the PDF $\phi(\cdot)$, computed over the summation of the cells V_i, with respect to the same integral computed over Q, namely:

$$\mathcal{A}(\mathcal{P}) = \frac{\sum_{i=1}^{n} \int_{V_i} \phi(q) dq}{\int_{Q} \phi(q) dq} \tag{3.2}$$

3.3.3 Overall Efficiency

Considering the fact that each robot is able to individually cover a circular area with radius r, we measure the effectiveness in the use of resources computing the ratio between the actual area covered by the team of drones, and the upper-bound on the area that could be theoretically covered, namely:

$$\mathcal{E}(\mathcal{P}) = \frac{\sum_{i=1}^{n} \int_{V_i} dq}{n \, \pi \, r^2} \tag{3.3}$$

3.4 Results and Analysis

In this section we report the results for both simulations and field tests performed according to the presented methodology. Quantitative results are related to the value of the optimization function defined in Equation (3.1), with larger values corresponding to

a configuration that is closer to the optimal one; to the integral of the PDF defined in Equation (3.2), where again larger values reflect better coverage efficiency; and last to the measurement of the overall covered area as defined in (3.3), where greater values reporting a better effectiveness in the usage of the robots.

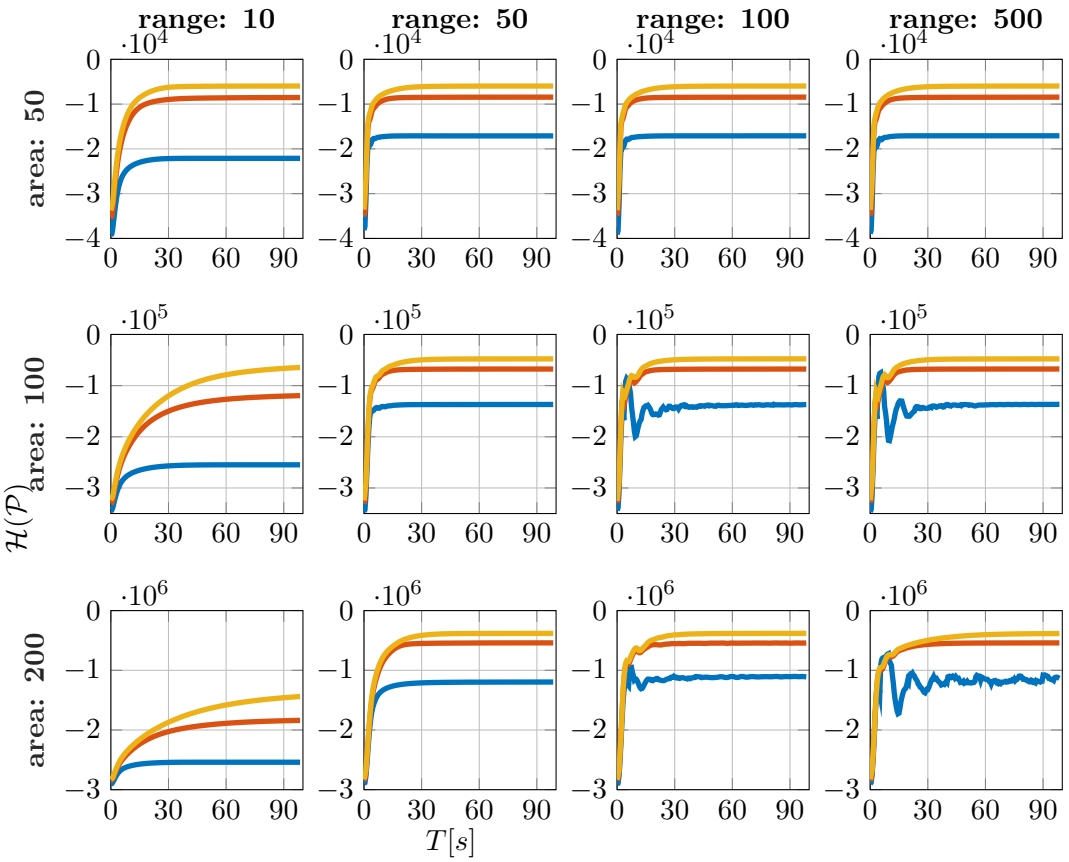


Figure 3.3: Average value of $\mathcal{H}(\mathcal{P})$ across all repetitions. The plots on the same line refer to the same total area to cover, while the plots on the same column refer to cases with same sensing range. The values for 8, 32 and 64 robots are displayed with blue, red and yellow lines respectively.

3.4.1 Simulations Results

We hereby present the results of the simulations. Figure 3.3 displays the average value of $\mathcal{H}(\mathcal{P})$ across the set repetitions. The blue line shows the value for 8 robots, the red line for 32 robots, and the yellow line for 64 robots. Each plot is organized according to the size of the area to cover and the sensing range. Following the same color scheme, Figures 3.4 and 3.5 display the average value of $\mathcal{A}(\mathcal{P})$ and $\mathcal{E}(\mathcal{P})$ respectively.

Figure 3.3 clearly shows that increasing the number of robots increases the performance with respect to the coverage metric. This effect is intuitively attributable to a greater cumulative covered area. It is also possible to note that the rate of convergence: (i) decreases with the increase of area size, (ii) increases if the sensing range grows, and (iii) decreases the more robots are employed. The oscillations present on the blue line for the cases with large Q and big sensing range are caused by the dynamics of the drones. Inherently, each drone possesses a certain motion dynamic different from $\dot{p}_i =$

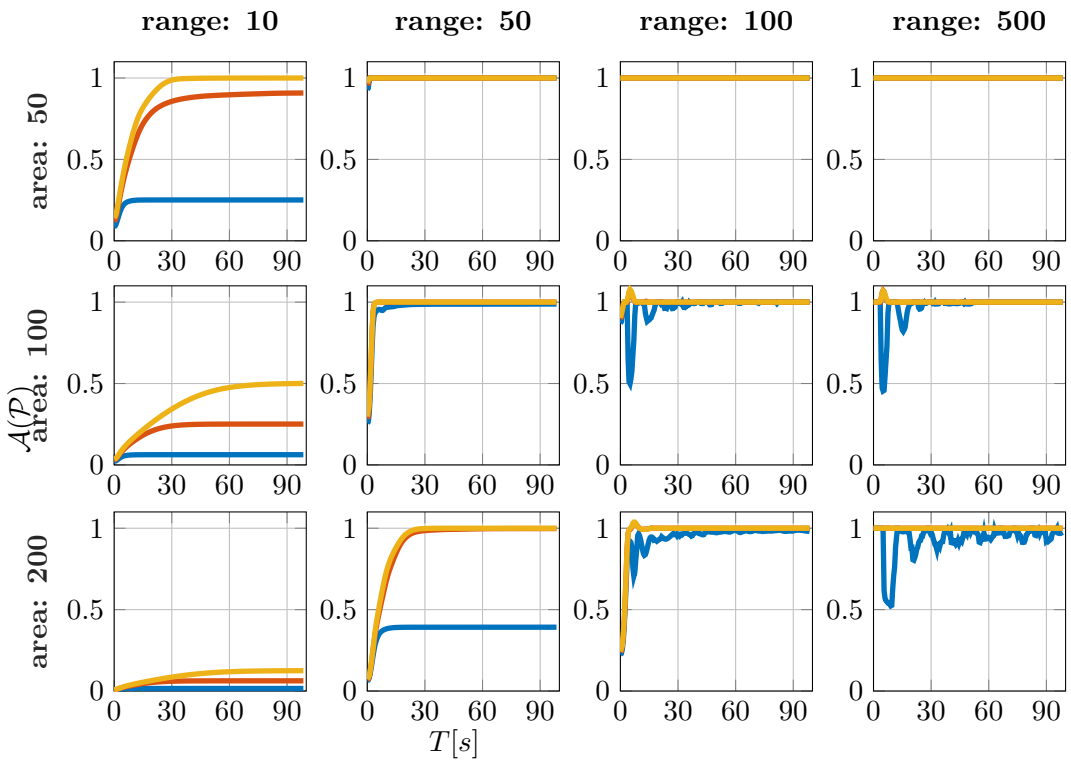


Figure 3.4: Average value of $\mathcal{A}(\mathcal{P})$ across all repetitions. The plots on the same line refer to the same total area to cover, while the plots on the same column refer to cases with same sensing range. The values for 8, 32 and 64 robots are displayed with blue, red and yellow lines respectively.

u_i. The velocity controller that generates the motor commands controls the drone so that its motion is as similar to (2.2) as possible. However, if the requested velocity is too different from the current velocity, the controller encounters acceleration limits that render it impossible to closely track the requested velocity profile. With large Q and large sensing range, the resulting V_i can be very large and their centroid can be far away from the drone. Therefore, the input generated by the control law (2.8) is very large, requiring accelerations outside the limitations of the drone, causing the oscillating behavior depicted by the blue line. With a larger number of drones, this behavior is limited or totally absent since each drone has a larger number of neighbors, therefore its resulting cell V_i is smaller. This effect can be reduced by reducing the value of k in (2.8) or by applying a saturation on the value of u_i. Nevertheless, the coverage algorithm is still capable of spreading drones across the area.

Figure 3.4 shows how well the area is covered by the drones. When the sensing range is small, the robots often struggle to cover the totality of Q. The covered area increases with the increase of the number of drones employed. This is expected since, obviously, more robots imply more cells V_i and the behavior for each drone is to distance itself from its neighbors, enlarging its cell V_i.

Figure 3.5 depicts the efficiency of the coverage. The plots clearly show that fewer robots are more efficient. A value of 1 for $\mathcal{E}$ means that each robot has reached the configuration where its cell V_i is the largest possible. With few robots there are fewer neighbors to interact with, ultimately allowing for the maximum enlargement of each

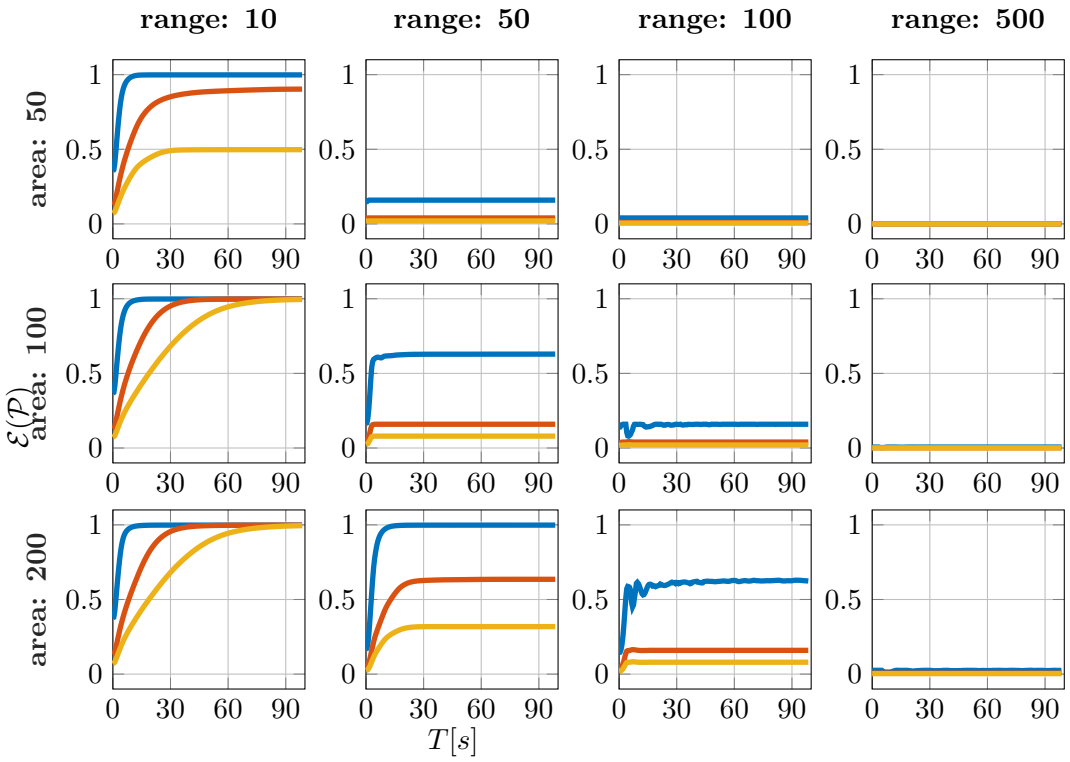


Figure 3.5: Average value of $\mathcal{E}(\mathcal{P})$ across all repetitions. The plots on the same line refer to the same total area to cover, while the plots on the same column refer to cases with same sensing range. The values for 8, 32 and 64 robots are displayed with blue, red and yellow lines respectively.

cell. The condition for a team of robots to not reach maximum efficiency that the total area Q cannot accommodate all the cells of the robot.

These considerations ultimately allow us to set some design analysis to make when deploying a coverage application. A larger number of robots is not necessarily better in all scenarios, even if it increases the performance. Depending on the sensing range, which is often dictated by different factors, a larger number of robots can imply a better area coverage but a trade-off has to be considered, in order to not overly reduce efficiency.

3.4.2 Field Experiments Results

We continue our analysis of the results with real field experiments involving a group of seven drones. Results are presented in Figure 3.6, where all the metrics detailed in Section 3.3 are utilized. In contrast to the simulated experiments, these tests are performed under realistic operating conditions, including sensing noise, communication delays, and actuation limits, which allow us to assess the practical behavior of the proposed coverage control strategy. The first line of graphs describes the metric $\mathcal{H}(\mathcal{P})$ for the different sensitivity range values taken into consideration, while the second and third lines report the metrics $\mathcal{A}(\mathcal{P})$ and $\mathcal{E}(\mathcal{P})$, respectively, providing complementary insights into coverage quality and efficiency.

Similarly to the simulated case, it can be observed that the metric $\mathcal{H}(\mathcal{P})$, shown in the first row as the blue curve, increases as the sensing range increases. This behavior is

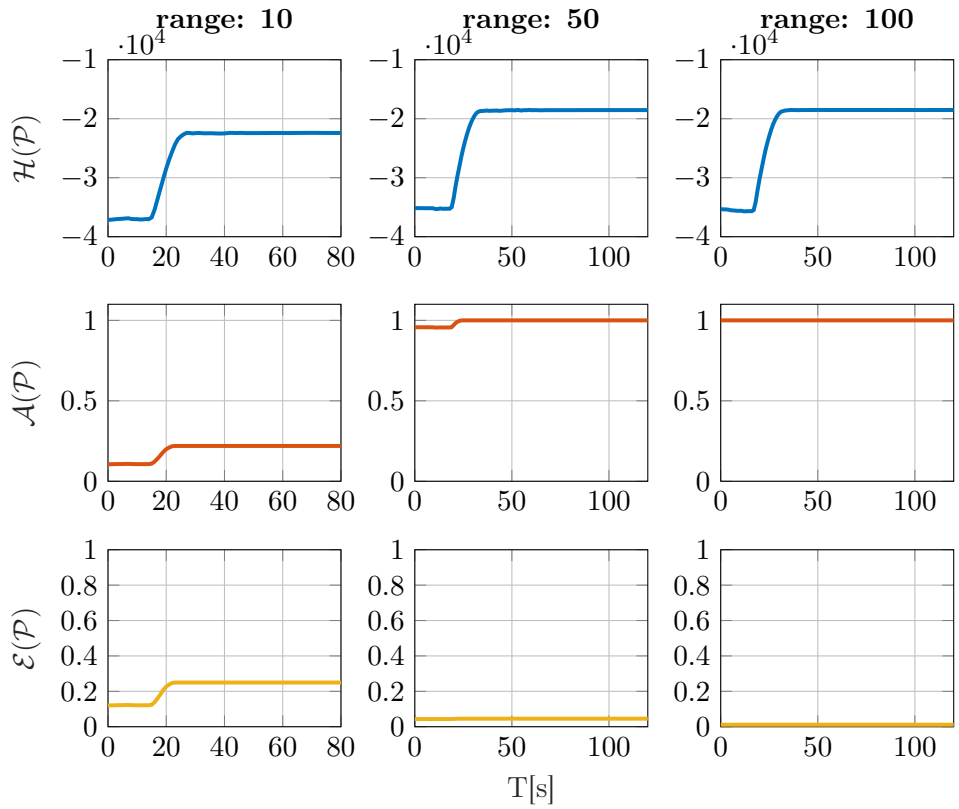


Figure 3.6: Average value of Equations (3.1), (3.2), (3.3) across all repetitions. The plots on the same line display the same metric, while the plots on the same column refer to cases with same sensing range.

consistent with the theoretical formulation of the coverage objective and confirms that larger sensing capabilities allow the system to achieve higher overall coverage performance. However, unlike the results reported in Figure 3.5, the convergence times of the optimization function defined in Equation (3.1) are noticeably longer. This difference is primarily due to the introduction of a maximum velocity cap of 2 m/s, which is enforced for safety reasons during real flight experiments and to prevent excessive overshoot or aggressive maneuvers on the physical platforms.

The impact of this velocity constraint becomes evident when comparing the results obtained for different values of the sensing range r. For $r = 10$, the drones are generally able to track the reference velocities generated by the proportional control law, resulting in relatively faster convergence. In contrast, for $r = 50$ and $r = 100$, the desired velocities frequently exceed the imposed limit, making the velocity cap particularly detrimental. In these cases, the vehicles are unable to reach the speeds commanded by the coverage controller, and the optimization function requires a longer time to settle to its steady-state value. Despite this slower convergence, the system remains stable and consistently approaches meaningful coverage configurations.

Figure 3.7 illustrates the initial and final conditions of the overall control process. Colored dots indicate the positions of the drones at the start (leftmost figure) and at the end (rightmost figure) of the experiment, while the circles surrounding each dot represent the sensing range of the corresponding robot. A visual inspection of Figure 3.7b reveals that the final configuration does not exhibit a compact and tightly packed formation.

Ideally, under noise-free conditions, the final arrangement would consist of adjacent but non-overlapping sensing regions; however, this ideal behavior is not fully achieved in the real experiments.

A closer analysis of the logged data provides insight into this effect. During the experiment, the GPS covariance terms $\sigma^2_{x,x}$ and $\sigma^2_{y,y}$ ranged between 1.0 and 1.4 meters, while the EPH (horizontal position error) and EPV (vertical position error) values oscillated between 0.6 and 0.95 meters. These levels of position uncertainty introduce non-negligible discrepancies between the estimated and actual drone positions, which in turn affect the computed Voronoi partitions and centroidal references. The resulting separation between sensing ranges is therefore consistent with the magnitude of the localization noise and does not indicate a failure of the coverage control strategy. Rather, it highlights the inherent limitations imposed by real-world sensing accuracy and confirms the robustness of the proposed approach under realistic operating conditions.

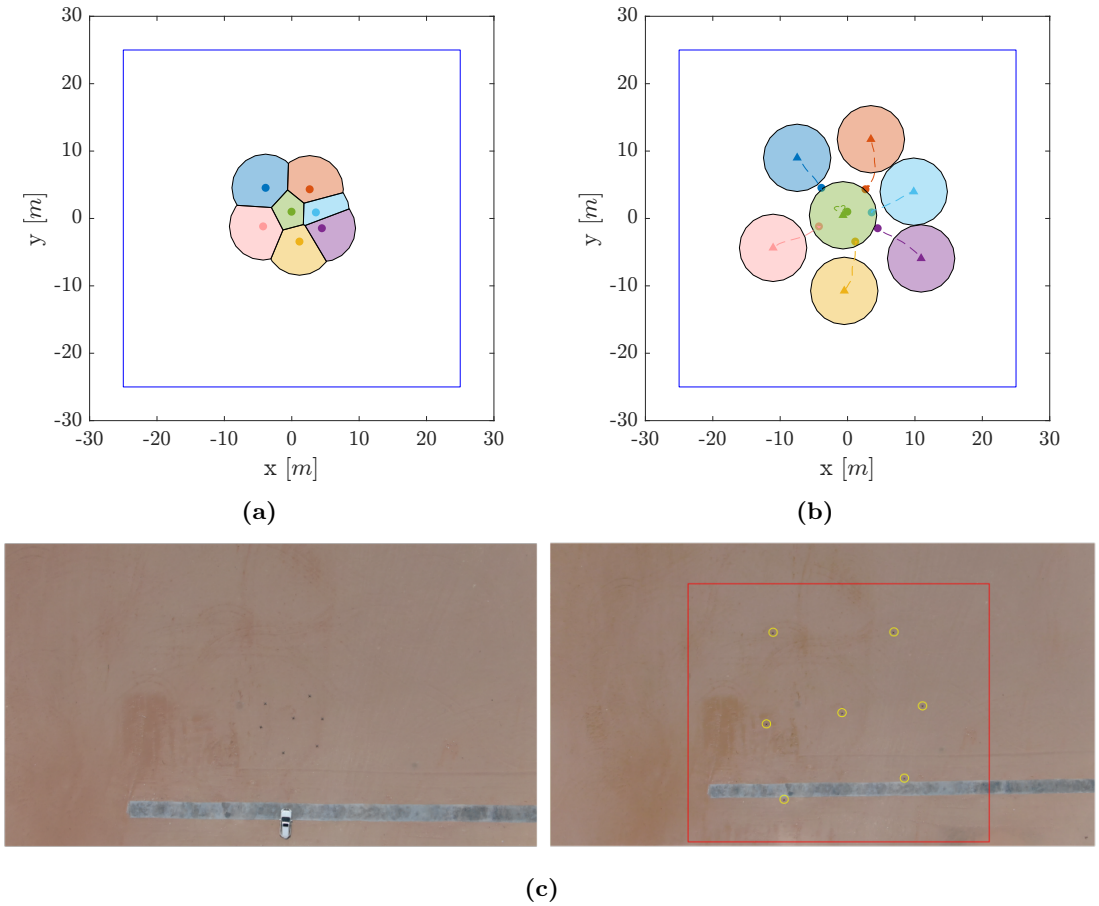


Figure 3.7: On image (a) the initial configuration of the vehicles during a real experiment with sensitivity range $R = 50$; on image (b), the final configuration of the drones in the same experiment. On image (c) an aerial photo of the convergence process is shown, with a different orientation of the axes.

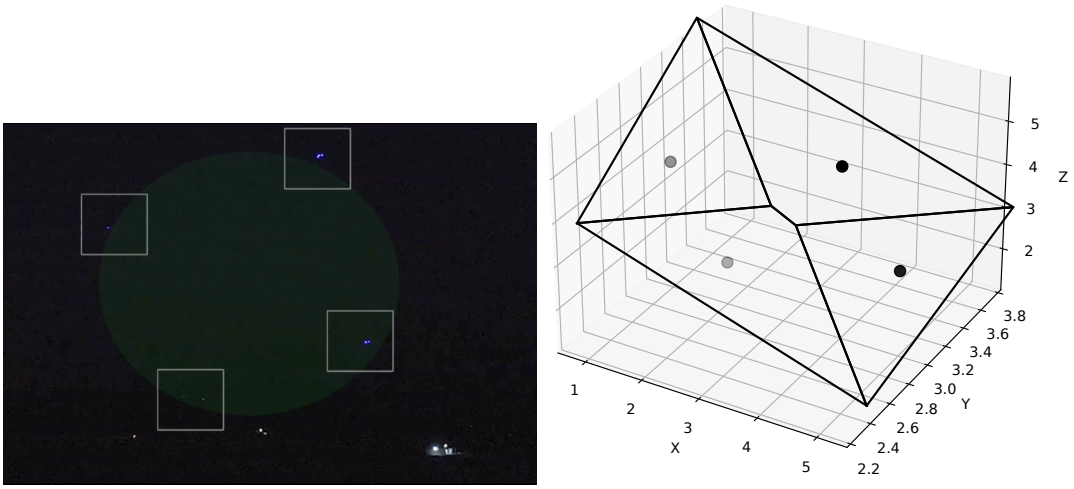
3.5 Discussions

In this part of the study, we evaluated a distributed coverage control algorithm capable of coordinating a group of autonomous aerial vehicles operating under limited sensing and communication capabilities. Through an extensive set of simulations and real-world flight experiments, the proposed strategy has been shown to be both robust and efficient. The controller consistently achieves close-to-optimal coverage configurations in both simulated and real environments, while maintaining stable behavior as the number of agents increases, thereby demonstrating the scalability of the approach.

Real flight experiments further highlight the impact of non-ideal operating conditions, such as noisy GPS measurements and actuation uncertainties, on the final coverage configuration. While these factors introduce small deviations from the ideal Voronoi partitions, they do not compromise the overall behavior or convergence properties of the coverage control algorithm.

An important aspect emerging from these results is the generality of the proposed formulation. Although the methodology is often illustrated using planar environments, the underlying control law and geometric construction are not restricted to horizontal surfaces. As shown in Figure 3.8, the same coverage control framework can be directly applied to vertical or arbitrarily oriented planar surfaces without modification. The Voronoi-based partitioning and centroid computation remain valid under a change of reference plane, demonstrating that the approach is inherently geometry-agnostic and can be extended to more complex surface geometries. This property can be particularly relevant for inspection use cases, monitoring, and surveillance tasks involving facades, walls, or other non-horizontal structures.

From a systems perspective, these results underline the suitability of the proposed coverage control framework as a reusable coordination primitive for aerial multi-robot applications. The reliance on local sensing, limited communication, and purely distributed computation makes the approach compatible with realistic deployment constraints, including scalability to larger teams and robustness to partial failures or intermittent communication losses. Moreover, the decoupling between the geometric coverage objective and the specific orientation of the underlying surface allows the same control architecture to be employed across a wide range of operational scenarios without re-design. This flexibility, combined with the experimentally demonstrated robustness of the controller, suggests that the proposed methodology can serve as a foundational building block for higher-level mission planning and task allocation strategies in real-world multi-UAV systems.



(a) **(b)**

Figure 3.8: Vertical coverage control example using the proposed framework. Four UAVs perform coverage on a vertical planar surface using the same distributed control law. Left: UAV positions during vertical coverage for a Gaussian density with $\sigma = 0.3$. Right: corresponding Voronoi partition.

Chapter 4

Time-varying and Adaptive Coverage Control

As discussed in the previous chapters, coverage control represents a fundamental coordination primitive for multi-robot systems, enabling a team of agents to distribute itself over a domain so that sensing or service effort is allocated where it is most valuable. Classical formulations, however, often rely on static environments, fixed team compositions, and idealized assumptions on sensing and communication. In real deployments, these assumptions rarely hold. Points of interest may drift, appear, or disappear; the operational environment may contain obstacles and unknown constraints; and the swarm itself may undergo changes in size, connectivity, or internal organization due to failures, mission updates, or task re-allocation. These challenges are particularly pronounced in aerial robotics applications such as environmental monitoring, surveillance, and search-and-rescue, where spatial information evolves on comparable timescales to the agents' motion and safety constraints are intrinsic to the task.

This Chapter investigates distributed coverage control under time-varying and adaptive conditions, focusing on how Voronoi-based methodologies can be extended beyond static settings to address realistic operational requirements. Spatial variability is modeled through PDFs defined over the workspace, allowing coverage objectives to capture both static and dynamic points of interest. Within this framework, agents rely exclusively on local sensing and limited-range communication, preserving scalability and robustness while avoiding global coordination assumptions. In addition to modeling spatial variability through probabilistic density functions, the proposed framework explicitly incorporates safety and coordination constraints at the control level. Control Barrier Functions are employed to enforce collision avoidance and obstacle avoidance requirements, while Control Lyapunov Functions are used to regulate formation and coordination objectives. This integration allows coverage optimization to be pursued in a constraint-aware manner, ensuring safe and structured execution under realistic operational conditions.

Two complementary aspects of adaptivity are studied. First, we analyze time-varying distributed coverage, where the emphasis is on the ability of a fully decentralized controller to track changes in the spatial distribution of points of interest while maintaining

high coverage quality. This includes continuous target motion, relocations, and variations in swarm size, all evaluated under realistic sensing and actuation non-idealities. Second, we address adaptive distributed coverage in constrained environments, where coverage objectives must coexist with safety, formation, and communication requirements. In this setting, coverage is embedded within a hierarchical control architecture that combines Voronoi-based optimization at the coordination level with control-theoretic mechanisms—specifically control barrier functions—to enforce collision avoidance, connectivity preservation, and task-related constraints.

Methodologically, the chapter builds upon the already introduced Lloyd's descent structure for coverage optimization while introducing practical extensions required for real-world deployment. These include smooth, state-dependent control gains to mitigate oscillatory behaviors observed in aerial platforms, reference-governor architectures compatible with standard SE(3) LLCs, and multi-layer control abstractions that decouple coverage objectives from safety-critical constraints. The proposed approaches are validated through experimental campaign results combining large-scale simulations and real-world quadrotor field trials.

The remainder of this chapter is organized as follows: Section 4.1 investigates time-varying distributed coverage control, analyzing the quality, robustness, and scalability of Voronoi-based strategies in the presence of dynamic points of interest and changing swarm configurations. Section 4.2 focuses on adaptive distributed coverage control, introducing a hierarchical framework that integrates coverage optimization with safety and formation constraints via control barrier functions, and evaluating its effectiveness in obstacle-populated and uncertain environments. Section 4.3 concludes the Chapter by summarizing the principal experimental results and discussing the key observations and insights gained from the proposed methodologies.

4.1 Time-Varying Distributed Coverage Control

4.1.1 Literature Review

In the classical setting, coverage is posed as a locational optimization problem in which agents iteratively move toward the centroids of their Voronoi partitions, yielding CVTs that are locally optimal with respect to a global coverage cost [1, 36]. Extensions to non-uniform spatial importance are commonly introduced through a weighting or density function, allowing agents to concentrate sensing effort in regions of interest while preserving distributed computation [24, 46, 47]. These concepts form the theoretical foundation of the decentralized Voronoi-based coverage framework introduced in Chapter 2, which serves as the baseline for the extensions considered in the remainder of this Section.

While these formulations provide strong convergence guarantees in static environments, many real-world scenarios are inherently dynamic. In environmental monitoring, surveillance, and disaster response, regions of interest may evolve over time due to moving

targets, spreading phenomena, or external disturbances. This motivates the study of *time-varying coverage*, where the optimal agent configuration is itself a moving target. Early approaches addressed this challenge by assuming slow temporal variations and relying on the inherent tracking capability of gradient-based coverage controllers [40, 48, 49]. However, such assumptions are often violated in aerial multi-robot systems, where actuation limits, sensing noise, and communication constraints interact with fast environmental changes.

A significant line of work has focused on learning or estimating unknown spatial importance online while performing coverage. Schwager *et al.* proposed a robust adaptive coverage framework in which robots actively explore the environment to learn an unknown weighting function and subsequently converge to locally optimal sensing configurations [50]. Their approach combines online function approximation with Voronoi-based control and provides formal guarantees on bounded learning error and convergence, even in the presence of approximation uncertainty. This work represents a foundational contribution to adaptive coverage, particularly in highlighting the role of robustness and exploration in distributed learning-based coverage.

Despite these advances, adaptive coverage controllers must ultimately address the control-theoretic challenges induced by continuously evolving spatial objectives. When the optimal configuration changes over time, the coverage law must track a moving reference while avoiding oscillations, overshoot, or loss of coherence among agents. In multi-UAV systems, these issues are further exacerbated by limited sensing ranges, intermittent communication, and the need to integrate coverage objectives with low-level flight control and safety constraints [29].

Recent experimental studies with aerial robot teams have demonstrated that classical Lloyd-type controllers, while theoretically sound, can exhibit undesirable transient behavior when deployed on real platforms, particularly in time-varying scenarios and under team-sizes variations [2, 47]. These observations motivate modifications to nominal coverage dynamics, such as bounded-speed references, smoothing mechanisms, and architectures that explicitly account for platform-level limitations. An initial analysis of time-varying targets in distributed coverage control was presented in previous work, highlighting the challenges induced by dynamic spatial objectives [3].

In summary, existing literature establishes Voronoi-based coverage as an effective distributed coordination strategy, and adaptive formulations provide mechanisms for learning unknown environments. However, the systematic study of *time-varying distributed coverage* in realistic multi-robot settings—where the environment, the swarm, and the sensing conditions evolve concurrently—remains limited. This gap motivates the focus of Section 4.1, which investigates the tracking capability, robustness, and experimental viability of distributed coverage control under explicitly time-varying spatial information.

4.1.2 Technical Approach

This Section presents the technical framework used to study distributed coverage control under time-varying spatial conditions. The approach is based directly on the decentralized, limited-range Voronoi-based coverage formulation introduced in Chapter 2. In particular, the definitions of Voronoi partitioning, centroidal Voronoi configurations, Lloyd-based descent dynamics, and the separation between coverage control and LLC vehicle stabilization are retained unchanged. The focus here is on how this framework is adapted to operate reliably when the coverage objective evolves over time and when deployed on real multi-UAV platforms.

We restrict attention to planar coverage at a fixed altitude $\bar{z}$, such that the coverage problem is formulated in $\mathbb{R}^2$ while the vehicles evolve in $\mathbb{R}^3$. This abstraction allows the analysis to isolate the behavior of the coverage layer, while maintaining compatibility with full six-degree-of-freedom flight dynamics.

In the presence of multiple targets, the density distribution is defined by combining the contribution of all detected or assigned targets. A convenient formulation consists in expressing $\phi(q,t)$ as a sum of localized components centered at the target positions, e.g., $\phi(q,t) = \sum_{\ell=1}^{N_T} w_\ell(t)\,\phi_\ell(q,t)$, where N_T is the number of targets, $\phi_\ell(q,t)$ is the spatial contribution associated with the ℓ-th target, and $w_\ell(t)$ is a weight that may encode its relative priority or confidence. In the considered framework, target positions may either be assigned externally by the operator, when the coverage objective reflects user-defined information, or estimated online by the UAV team through the cooperative triangulation process described in this section. Therefore, the same coverage architecture can accommodate both operator-driven and autonomous target selection.

4.1.2.1 Time-Varying Coverage Objective

As in Chapter 2, spatial importance is modeled through a PDF $\phi(q)$ defined over the environment $Q \subset \mathbb{R}^2$. In contrast to the static scenarios previously considered, here the density is allowed to vary explicitly in time, yielding a time-dependent coverage objective $\phi(q,t)$. This formulation captures a wide range of realistic situations in which points of interest may move, appear, disappear, or change intensity.

The Voronoi partition $\mathcal{V}(P)$ continues to be generated from the instantaneous agent positions $P = \{p_1, \ldots, p_n\}$, and the centroid of each cell is computed according to the weighted definition introduced in Section 2.2.3. However, when $\phi(q,t)$ varies in time, the centroid of a Voronoi cell becomes a moving target even when the partition geometry remains unchanged. As a result, the classical notion of convergence to a static centroidal Voronoi configuration is replaced by a tracking problem, in which agents must continuously follow time-varying reference points.

Dynamics of Moving Centroidal Set-Points The Lloyd's descent dynamics reviewed in Chapter 2 guarantee convergence to a locally optimal configuration under static conditions. When the density $\phi(q,t)$ evolves, the same dynamics induce a trade-off

between responsiveness and stability. Aggressive tracking of rapidly moving centroids can lead to oscillations and loss of coherence, while overly damped responses degrade coverage quality by introducing significant lag with respect to the evolving spatial information.

This effect is particularly pronounced in aerial systems, where sensing noise, actuation limits, and asynchronous updates interact with the motion of the centroidal references. Empirical observations from field experiments indicate that the proportional Lloyd's controller adopted in Chapter 2 is insufficiently robust under these conditions, motivating a modification of the control law at the coverage layer.

4.1.2.2 Modified Lloyd's Control Law

In Chapter 2, the coverage control input is generated using a proportional controller that drives each agent toward the centroid of its Voronoi cell. While effective in simulation and static environments, this approach was observed to produce overshoot and sustained oscillations when applied to real UAV platforms operating under time-varying coverage objectives, as seen in Section 3.4.

To address this limitation, we introduce a state-dependent nonlinear gain that modulates the Lloyd's descent dynamics as a function of the distance to the centroid. Specifically, the proportional gain is replaced by a smooth gain $k_s(e_i)$, where $e_i = \|C_i - p_i\|$ denotes the distance between the agent and its assigned centroid. The gain is defined as the superposition of two sigmoidal functions:

$$k_s(e_i) = \frac{1}{1 + e^{(e_i/s+s)}} + \frac{1}{1 + e^{-(e_i/s-s)}}, \tag{4.1}$$

where the parameter $s \in \mathbb{R}$ controls the sharpness of the transition.

This construction yields low control authority near the centroid, reducing sensitivity to noise and discretization effects, while preserving sufficiently large gains at greater distances to ensure responsiveness. The resulting coverage control law is given by:

$$\dot{p}_i = u_i = k_s(e_i)\,(C_i - p_i), \tag{4.2}$$

and retains the decentralized structure of the original Lloyd's controller. Similar nonlinear gain structures have been shown to improve stability in real-world multi-robot deployments [51, 52], and their effectiveness in the present context is evaluated experimentally in Section 4.1.5.

4.1.2.3 Integration with Low-Level Flight Control

The coverage controller generates position set-points that are passed to a LLC responsible for enforcing vehicle dynamics and safety constraints. As described in Chapter 2, we employ a geometric controller defined on the special Euclidean group SE(3), which computes thrust and attitude commands to track the desired position references while stabilizing the vehicle orientation.

Importantly, the coverage layer remains agnostic to the details of the LLC. This separation enables the proposed strategy to be applied across different robotic platforms, provided that a suitable reference-tracking controller is available. In this work, the SE(3) controller serves as a reference governor that ensures compatibility between the time-varying coverage objectives and the physical limitations of quadrotor UAVs.

In summary, the technical approach of this chapter reuses the decentralized Voronoi-based coverage framework developed in Chapter 2, while extending it to explicitly address time-varying spatial information and real-world deployment constraints. The primary modification consists in reformulating coverage as a tracking problem and introducing a nonlinear gain in the Lloyd's descent dynamics to improve stability under dynamic conditions. The impact of these design choices is analyzed quantitatively through simulations and validated experimentally in the following Sections.

4.1.3 Perception in the loop

To enable time-varying coverage with real targets, the spatial knowledge $\phi(q,t)$ is estimated online from onboard perception and multi-view geometry. Each UAVs produces object detections from its forward-looking camera and converts them into a LoS observation expressed in the world frame. These observations are shared among vehicles and fused through a triangulation module to obtain a global target estimate and an associated uncertainty indicator. The resulting estimate is then mapped to the density function $\phi(q,t)$ used by the coverage controller.

LoS observation model Let ${}^{W}T_{C_i}(t)$ denote the pose of the sensor frame of UAV i—i.e. in our case, a camera sensor—with respect to the world frame, obtained from the onboard state estimator and the camera extrinsic parameters. Given a detection in the image, the pixel coordinates of the bounding-box center are back-projected through the camera model to obtain a unit line-of-sight direction vector expressed in the camera frame.

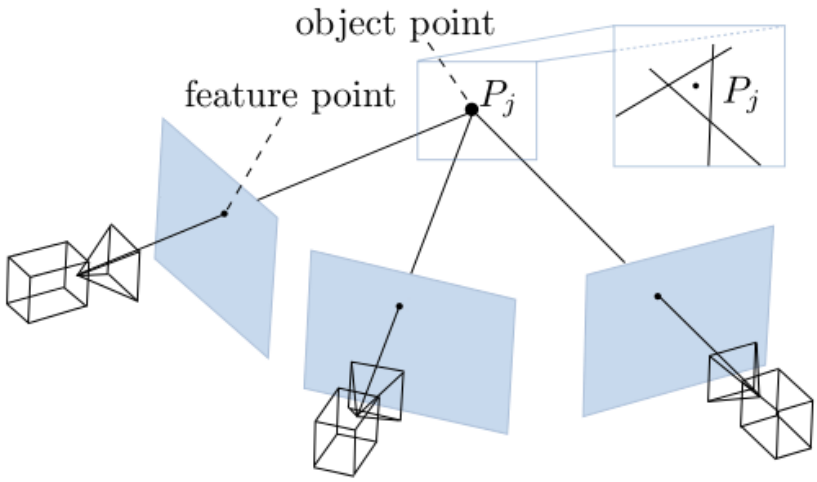


Figure 4.1: Example of triangulation process with different point if views.

By applying the camera-to-world transform, the LoS direction expressed in the world frame is:

$$ {}^{W}b_i(t) = {}^{W}R_{C_i}(t)\, {}^{C_i}b_i(t), \qquad \|{}^{W}b_i(t)\| = 1. \tag{4.3} $$

Equivalently, the observation can be represented by azimuth and elevation angles with respect to the world frame,

$$ \alpha_i(t) = \operatorname{atan2}(b_{i,y}(t),\, b_{i,x}(t)), \qquad \beta_i(t) = \operatorname{atan2}(b_{i,z}(t),\, \sqrt{b_{i,x}^2(t) + b_{i,y}^2(t)}), \tag{4.4} $$

together with the sensor position ${}^{W}p_{C_i}(t)$.

Multi-view triangulation and confidence At each time step, the set of available measurements from multiple UAVs is fused to estimate the 3D target position $\hat{x}(t) \in \mathbb{R}^3$. We formulate triangulation as a nonlinear least-squares problem: the estimated target position should minimize the aggregate discrepancy between the predicted LoS angles from each sensor and the measured angles. Denoting a measurement as:

$$ z_i(t) = \big({}^{W}p_{C_i}(t),\, \alpha_i(t),\, \beta_i(t)\big), \tag{4.5} $$

the estimator computes:

$$ \hat{x}(t) = \arg\min_{x \in \mathbb{R}^3} \sum_{i \in \mathcal{I}(t)} \left\| \begin{bmatrix} \alpha(x; {}^{W}p_{C_i}) - \alpha_i \\ \beta(x; {}^{W}p_{C_i}) - \beta_i \end{bmatrix} \right\|^2, \tag{4.6} $$

where $\mathcal{I}(t)$ is the set of UAVs providing valid detections at time t. In implementation, Equation (4.6) is solved via a Levenberg–Marquardt procedure [53], and the final residual norm is used as a confidence indicator for the estimate (lower residual implies better geometric consistency).

Mapping to a time-varying density The estimated target location is projected onto the coverage plane at altitude $\bar{z}$ to obtain the mean of the coverage density, i.e.,

$$ q_0(t) = \Pi_{\bar{z}}(\hat{x}(t)) \in Q \subset \mathbb{R}^2, \tag{4.7} $$

and the information is modeled as a Gaussian density

$$ \phi(q, t) = A(t) \exp\left(-\frac{\|q - q_0(t)\|^2}{2\sigma^2(t)} \right). \tag{4.8} $$

In our experiments, $\sigma(t)$ is selected as a monotone function of the triangulation confidence (residual norm), yielding larger uncertainty for poorer multi-view agreement and

tighter peaks when triangulation is well-conditioned. This closes the loop from perception to coverage control, enabling the team to react to moving targets using purely onboard sensing and distributed information exchange. For a details analysis on projective transformation methodologies we refer to [54].

4.1.4 Experiments

Figure 4.2: Figure of the quad-rotor platform selected for the experiments

To fully validate our approach, the proposed algorithm has been tested on a group of UAV both in simulation and in real-world field tests. We focus on operating the group in vast outdoor setting with sparse time-varying feature expressed as PDF. All the experiments are evaluated using the metrics reported in Section 3.3.

Onboard detection and target localization: implementation Each UAV runs an onboard object-detection pipeline (YOLO family) [55] to infer 2D bounding boxes from the forward-looking camera stream. For each detection, the pixel center of the bounding box is converted to a LoS bearing using the calibrated camera intrinsics, and transformed into the world frame using the UAV state estimate and the camera extrinsics. Each vehicle broadcasts its observation (sensor pose and LoS angles) to its neighbors. A dedicated ROS 2 triangulation node fuses multi-UAV observations by solving a nonlinear least-squares problem via Levenberg–Marquardt and publishes the estimated target position together with a confidence score. The coverage controller then constructs the time-varying density $\phi(q, t)$ as in (4.8).

Set of experiments Simulated experiments are categorized into three main categories:

1. Multiple static targets,
2. Single time-varying target,
3. Single target that varies discretely and non-continuously over time.

Real-world experiments focus on the same three fundamental scenarios under consideration and, even if less in number, show more complex scenarios with real targets and full

vision in the loop—i.e.; to generate the PDF—allowing us to demonstrate the robustness of the proposed approach in the presence of sensor noise and unpredictable challenges. The control algorithm, detailed in Section 2.2, has been implemented utilizing the C++ version of the ROS framework both for real and simulated experiments. In the former, the full Software In The Loop (SITL) has been reproduced on individual threads assigned to each UAV. Each thread handles the coverage control, the LLC for quad-rotor flight implementing the reference governor architecture, information sharing, and the PX4 autopilot. To ensure the independence from initial conditions, vehicles are spawned in the environment with a random displacement around the coverage area and different seeds have been used across different runs to do so. Real-world experiments were conducted in a remote area located in the Al Mirfa region of the Abu Dhabi emirate in the United Arab Emirates. The aerial platform of choice is a custom-designed quad-rotor developed at the Technology Innovation Institute (TII) which comprises a 60 cm carbon frame, a Pixhawk4 6C or 6X autopilot, and an NVIDIA Jetson Xavier NX as the on-board computer responsible for the mission and flight control stack. Communication among different drones relies on radio frequency with a carrier frequency of 5.8 Ghz. Other sensors include a GPS module for self-localization, a short-range Lidar for short-range altitude estimation and a front-facing stripped-down IMX77 camera for feature localization. For safety in field trials, the commanded ground speed induced by (4.2) is saturated to a maximum of 8 m/s.

Camera pointing and yaw control To maintain the target within the camera field of view, the heading of each UAV is controlled independently from the translational coverage dynamics. The desired yaw $\psi^\star(t)$ is computed from the current target bearing in the horizontal plane, and tracked using a PID controller acting on the yaw error $e_\psi(t) = \text{wrap}(\psi^\star(t) - \psi(t))$, where $\text{wrap}(\cdot)$ denotes an angle normalization operator that maps angular quantities to the principal interval $(-\pi, \pi]$. This operation ensures that the yaw error is expressed as the smallest signed angular difference between the desired and measured headings, avoiding discontinuities due to the periodic nature of angular variables and preventing spurious large control actions when crossing the $\pm\pi$ boundary. This heading stabilization improves detection continuity, increases triangulation observability, and reduces missed detections during aggressive target motion. While the coverage dynamics are formulated on $\mathbb{R}^2$ (Section 4.1.2), the implemented command interface extends the reference with yaw-rate/heading tracking.

4.1.5 Results

To evaluate the system performance, we rely on identical metrics as the one used in [2]:

1. *Configuration optimality* $\mathcal{H}(\mathcal{P})$, as defined in Equation (3.1)
2. *Coverage effectiveness* $\mathcal{A}(\mathcal{P})$, as defined in Equation (3.2)
3. *Overall efficiency* $\mathcal{E}(\mathcal{P})$, as defined in Equation (3.3)

To briefly summarize, in (3.1) we aim at evaluating how close the configuration of the robots is to the centroidal Voronoi configuration, defined considering for each point $q \in Q$ the value of the PDF $\phi(q)$. Next, (3.2) computes the *coverage effectiveness* and shows how well the environment Q is covered by non-overlapping individuals with limited-range cells. Last, (3.3) is expressed as the effectiveness in the use of resources and is done by computing the ratio between the actual area covered and the upper-bound on the area that could be theoretically covered. Here r is the drones range and n the number of agents which is the same as the number of polygons composing $\mathcal{V}$.

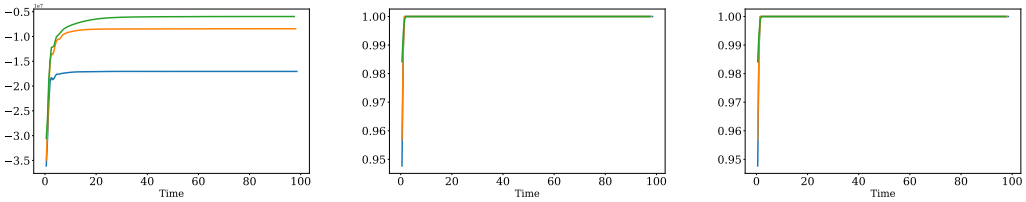
Results of simulations Figure 4.3 shows the average value of $\mathcal{H}(\mathcal{P})$, $\mathcal{A}(\mathcal{P})$ and $\mathcal{E}(\mathcal{P})$ across the set repetitions in the context of the *single time-varying target* experiment. The blue line shows the value for 6 robots, the orange line for 12 robots, and the green line for 24 robots. Following the same color scheme, Figures 4.4 and 4.5 display the average value of $\mathcal{A}(\mathcal{P})$ and $\mathcal{E}(\mathcal{P})$ for the *multiple static targets* and the *single target that varies discretely and non-continuously* experiments respectively. In all simulated experiments, a coverage area of 200×200 meters is employed, with a range value of $r = 200$ meters. The selection of this variable is empirically set to be consistent with the communication stack used by the drones, thereby allowing visibility between drones. By analysis of the results in Figure 4.3, in the case of a *single time-varying target*, we notice that the optimality parameter $\mathcal{H}(\mathcal{P})$ exhibits a behavior compatible to a static scenario. This can be rationalized by the fact that the robots tend to pursue the target while maintaining a consistent maximization of this metric over time. Even though the PDF $\phi(q, t)$ varies, the pursuit of the target tends to mitigate these variations keeping $\mathcal{H}(\mathcal{P})$ at its maximum.

On the other end, Figure 4.4, shows that for the context of *multiple static targets*, the behavior aligns and confirms intuitive expectations: After a initial phase, the values of all three metrics reach their steady-state maxima. It can be noted that the transients of the functions $\mathcal{H}(\mathcal{P})$ and $\mathcal{A}(\mathcal{P})$ persist for a longer duration with respect to the previous case. When compared to the latter, we note that when multiple targets are set, the drones take more time to reach an equilibrium state that maximizes the values of the two functions.

Last, in Figure 4.5 we show the *single target that varies discretely and non-continuously* scenario. It is interesting to observe how the two metrics $\mathcal{H}(\mathcal{P})$ and $\mathcal{A}(\mathcal{P})$, experience reductions in their values following the achievement of a stable equilibrium. These reductions are a consequence of the discrete and non-continuous changes in the target's position—represented by the mean value of $\phi(q, t)$. However, the metrics subsequently return to the equilibrium values that maximize their functions. This outcome is noteworthy as it demonstrates the robustness of the approach in the face of unexpected, unforeseen variations in the environment.

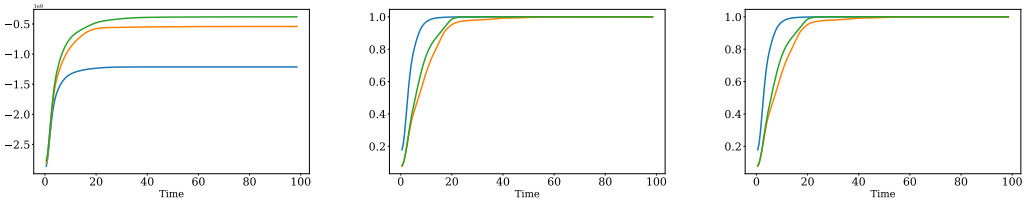
Across all three scenarios, the value of overall efficiency $\mathcal{E}(\mathcal{P})$, remains constant over time. This constant value is an inherent result of the drone range, which allows for a continual and consistent overlapping of the coverage area.

Figure 4.6 illustrates the function described by Equation (4.1) for different values of s.



(a) $\mathcal{H}(\mathcal{P})$ **(b)** $\mathcal{A}(\mathcal{P})$ **(c)** $\mathcal{E}(\mathcal{P})$

Figure 4.3: Results for the single time-varying target case experiments.

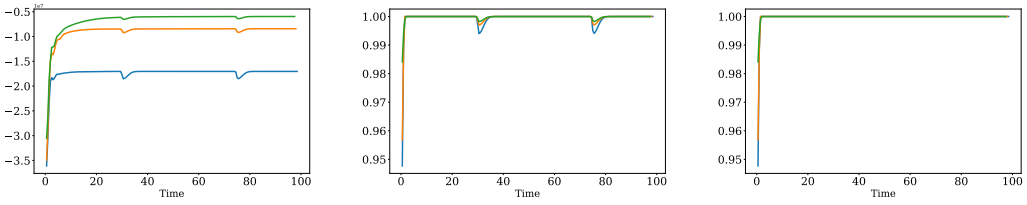


(a) $\mathcal{H}(\mathcal{P})$ **(b)** $\mathcal{A}(\mathcal{P})$ **(c)** $\mathcal{E}(\mathcal{P})$

Figure 4.4: Results for the multiple static targets case experiments.

Figure 4.7 demonstrates the effect of the application of Equation (4.2), this function reduces the value of the first-order control for small values of $(C_i - p_i)$ on the two components of control, x and y compared to not using this approach. The stabilization is evident from the fact that the transient response exhibits fewer oscillations caused by proportional control overshoot. The selection of value $s \in \mathbb{R}$ has a significant impact on the overall behavior, and it depends on the size of the area and the interactions among nearby drones—distance from neighbors.In Section 3.4, it has been demonstrated that drone movements can exhibit strong oscillations during the transient response toward the new computed point. Here, results shows how this phenomenon is attenuated by the control law in (4.2).

Results of field tests In Figure 4.8 (right), the target is shown at two different instants of its motion: the initial position and the final position. The two red circles denote the target-centered regions associated with these two states, highlighting how the area of interest, and consequently the density function—moves over time together with the target. In this experiment, the target locations are assigned externally in the first phase, representing user-defined areas of interest used to construct the density map; in the second phase the target locations used to generate the density map are assumed to be available from the cooperative triangulation process described above. Figure 4.9 shows the results of the three metrics for the real experiment scenario. Apart from the higher noise in the $\mathcal{H}(\mathcal{P})$ metric, which is strongly tied to the quality of the drone's position information, the positions of neighboring drones, and the number of repetitions made for the measurement, the same conclusions as the simulated experiments can be inferred. The instantaneous change in $\phi(q, t)$ does not destabilize the behavior and allows the system to regain equilibrium following a transient event. A greater number of vehicles maximizes the value of $\mathcal{H}(\mathcal{P})$, even though it may not be an optimal choice in all scenarios.



(a) $\mathcal{H}(\mathcal{P})$ **(b)** $\mathcal{A}(\mathcal{P})$ **(c)** $\mathcal{E}(\mathcal{P})$

Figure 4.5: Simulation results for single target discrete-variation case and for non-continuously over time experiments.

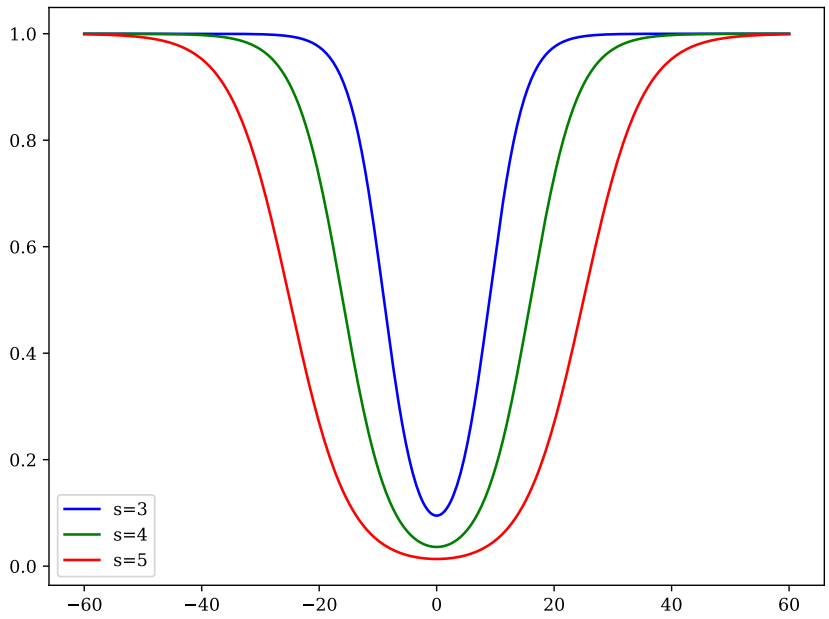


Figure 4.6: Visualization of the function used to harmonize the dependence of the gain k_s on the difference $(C_i - p_i)$. Shape of the function for different values of s

4.1.6 Discussion

Figure 4.10 provides a comprehensive synthesis of the real-world experiment, illustrating the tight integration of onboard perception, inter-UAV communication, distributed triangulation, and coverage control. Each drone is equipped with an object-detection pipeline that enables the recognition of the white vehicle, while the combined action of yaw control and the distributed triangulation process described in Section 4.1.3 allows the swarm to estimate the position of a time-varying target in real time. This estimated target location is then used to construct the dynamic coverage objective, closing the perception–control loop and enabling adaptive coverage behavior in a realistic deployment scenario.

The results analyzed demonstrated robustness in the face of temporal environment variations by dynamically adapting the coverage strategy based on real-time information: it effectively responded to changes such as moving targets. This adaptability ensured reliable

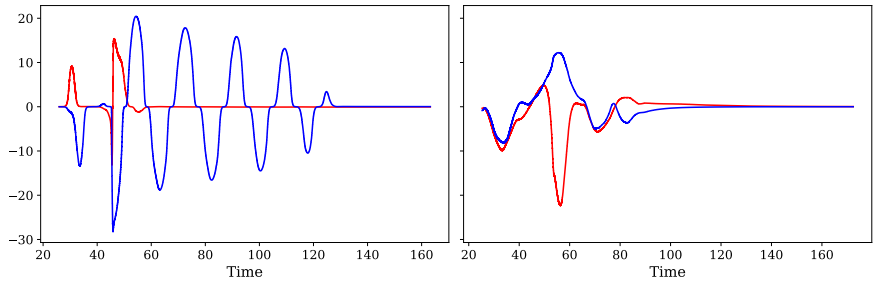


Figure 4.7: Visual representation over time of the first-order control function with (right) and without (left) the overlapping sigmoid functions.

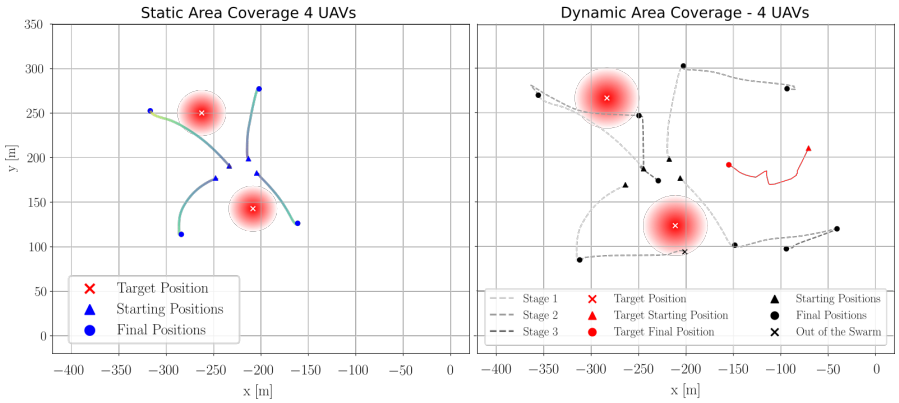
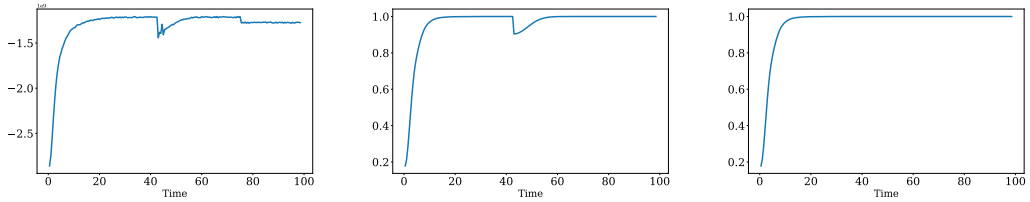


Figure 4.8: Experimental sequence of the adaptive clustered coverage framework. Left: the UAV swarm takes off and the coverage process starts from two initial static points of interest (POIs), which define the initial areas of interest to be monitored. Right: after this initial phase, the mission switches to a moving non-static target, whose trajectory is indicated by the red line. During this stage, one UAV loses communication with the swarm and performs a landing procedure due to a fault. The remaining UAVs then autonomously reconfigure the swarm and continue the coverage task with the reduced team, preserving the overall collective behavior.

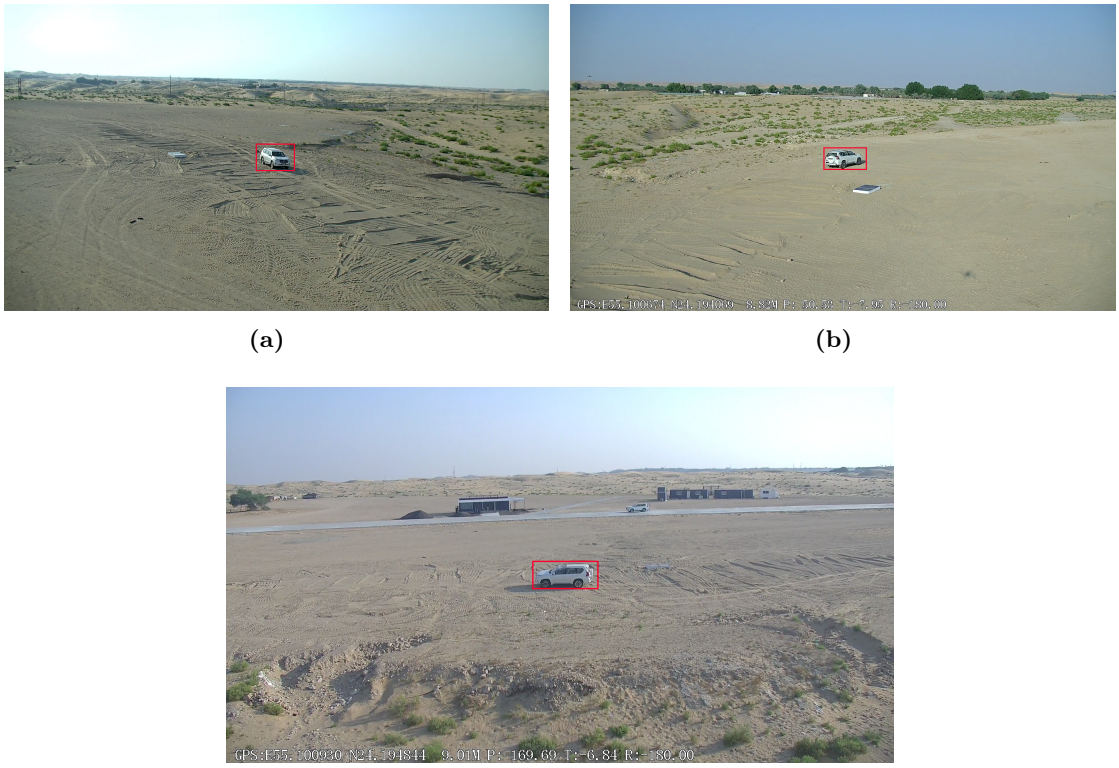
and efficient coverage performance in dynamic and unpredictable scenarios. Moreover, the algorithm exhibited scalability with respect to the number of drones in the original swarm, and our approach efficiently managed the coordination and cooperation across different swarm sizes, enabling seamless scalability and consistent coverage quality. Our algorithm showcased robustness to changes in swarm conditions during operations, it effectively handled variations in the number of drones which differ from the original swarm size. The presence of faulty or malfunctioning drones, and the addition or removal of drones during operation does not effect the algorithm, and by employing adaptive control mechanisms and swarm reconfiguration strategies, the algorithm ensured the continuity and stability of coverage performance. Additionally, we introduced a novel control law that effectively mitigated oscillations in the coverage trajectories. Unlike traditional proportional control laws that may lead to oscillatory behaviors, our proposed control law incorporated smoothing techniques, resulting in gentle and stable coverage paths. This improvement significantly enhanced the overall performance of the coverage algorithm and ensured consistent and precise coverage without undesired oscillations.

In conclusion, our experiments demonstrated the effectiveness and robustness of the time variant coverage control algorithm, its ability to handle temporal variations, its scalability with respect to the number of drones, the robustness to changes in swarm conditions,



(a) $\mathcal{H}(\mathcal{P})$ **(b)** $\mathcal{A}(\mathcal{P})$ **(c)** $\mathcal{E}(\mathcal{P})$

Figure 4.9: Results for the real-world experiment.



(a) **(b)**

(c)

Figure 4.10: View of the drone cameras during the real experiment..

and the effective mitigation of oscillations. All the above are valuable contribution to the field of swarm robotics and autonomous systems with a particular accent to real-field deployment. We believe that these results could pave the way for further research and practical applications in various domains.

4.2 Adaptive Distributed Coverage Control

4.2.1 Introduction

Building on the decentralized coverage framework introduced in Chapter 2 and the time-varying perception driven coverage strategy presented in Section 4.1, this Section addresses the problem of enforcing safety, keeping formation and communication constraints during coverage execution. While Section 4.1 focused on tracking dynamic spatial information through a time-varying density function $\phi(q, t)$, here we study how coverage objectives can be pursued in the presence of environmental obstacles, inter-agent con-

straints, and sensing limitations by explicitly embedding constraint handling into the control design.

These considerations impact in a broad range of aerial multi-robot applications, including exploration [56], precision agriculture [11], and target monitoring/tracking [57, 58]. In this Section, we focus on a sensing-limited setting that is representative of lightweight and low-cost deployments: each UAV is equipped only with a monocular camera and must detect and localize events of interest while operating under safety and coordination constraints. Since monocular detections do not directly provide depth, target localization requires either motion-induced parallax or multi-view geometric fusion; here we explicitly leverage cooperation among UAVs to enable localization while maintaining safe and structured team behaviors.

The solution we propose aims at exploiting cooperation among UAVs to monitor the environment. Specifically, we define sub-groups of the entire team, each one composed by n UAVs capable of collectively localizing a target combining their image data. The control architecture is designed as a combination of formation control for coordination of robots within the same sub-group, and coverage control for coordination among the whole team. In the following sections, we describe how a two-level semi-centralized controller is designed to achieve the mentioned behavior, and we present the experimental evaluation we conducted to test the effectiveness of the solution.

To summarize, the contributions of this Section are:

- A formation controller based on Control Barrier Functions (CBFs) and Control Lyapunov Functions (CLFs), enforcing inter-UAV safety, obstacle avoidance, and intra-cluster connectivity while stabilizing a camera-informed formation.
- A cooperative coverage strategy acting at the cluster level, which allocates sensing effort over a density map $\phi(q, t)$ by steering cluster virtual agents, thereby enabling multi-view localization of events of interest using monocular cameras.

It is important to emphasize that this section does not introduce a new coverage control law. The Voronoi-based coverage formulation and Lloyd-type descent dynamics are inherited directly from Chapter 2, while the perception-driven, time-varying density generation is addressed in Section 4.1 and it is still valid. The focus of this section is instead on embedding such coverage objectives within a constraint-aware control architecture, enabling their execution in the presence of inter-agent safety constraints, limited sensing, and environmental obstacles. In this sense, the proposed framework complements the previous sections by addressing the practical feasibility of coverage control in realistic operational conditions.

In contrast to the coverage formulations presented in Chapter 2 and Section 4.1, where individual robots act directly as coverage agents, this Section introduces a hierarchical abstraction: Coverage objectives are assigned to virtual agents associated with robot clusters, while physical UAVs are responsible for executing these objectives subject to sensing, safety, and communication constraints. This abstraction preserves the underlying Voronoi-based coverage formulation, while enabling constraint-aware execution in realistic environments.

4.2.2 Related Works

4.2.2.1 Monocular and Cooperative Target Tracking

The problem of target tracking with a monocular camera has been addressed in [59], where authors propose a solution to retrieve the position of a target based on pinhole imaging, allowing to map three-dimensional objects to a two-dimensional plane, in combination with a geometric model to find the relationship between pixels in the image and points in the real world. Instead, the authors in [60] propose a target localization strategy to retrieve depth data from sequential images. By exploiting an Extended Kalman Filter for robot localization, this solution allows to triangulate the target's position by observing it repeatedly from different spots. A different approach has recently emerged with the use of Deep Learning techniques, as pointed out in [61]: recent considerable progress in learning-based techniques, specifically Computer Vision, has brought a wide range of solutions to the problem of depth estimation, typically outperforming traditional solutions. Early investigations on UAV-based target tracking using monocular vision highlighted the inherent limitations of single-view geometry [62], cooperative target tracking approaches for multi-robot systems relied on probabilistic filtering techniques to fuse distributed observations [63]. Joint localization and target tracking from monocular imagery has been investigated as a means to overcome depth ambiguity through motion and estimation coupling [64]. An interesting approach involves cooperation among robots to get the target's position. In [65] a model predictive control-based decentralized controller is proposed that brings robots to converge around the target while avoiding static and dynamic obstacles. In addition, learning-based methods have been also applied to cooperation settings. An example is the solution in [66], which leverages Graph Neural Networks to select the control action from local observations and communication only, thus making it suitable for large-scale scenarios.

4.2.2.2 Formation Control

Formation control is another extensively studied topic within the multi-robot research community. A distributed solution is proposed in [67] to find collision-free trajectories via constrained optimization, allowing for reconfiguration. A similar approach is also used in [68] for collaborative object transport. Artificial potential fields can also be employed for this purpose. An example is [69], where their employment allows a multi-robot system to be shaped as an arbitrarily defined polygon. Finally, already mentioned Deep Learning has also been applied to this scenario to train robots to organize into formations while avoiding collision with other agents and obstacles [70].

4.2.3 Preliminaries

In this Section, we describe the considered scenario and present the assumptions we make, then we briefly present the tools used in the controller's design.

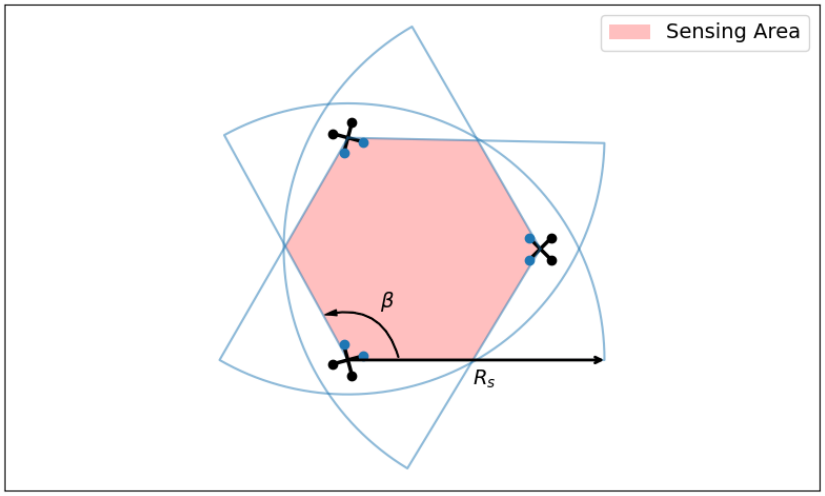


Figure 4.11: Sensing model: the sensing area of the cluster results from the intersection of each UAV's field of view.

4.2.3.1 Problem Description

Consider a team of $N \in \mathbb{N}$ aerial robots flying at the same constant altitude $\overline{z} \in \mathbb{R}^+$, thus moving in a 2D environment $Q \subset \mathbb{R}^2$ possibly populated with obstacles. The notation $q \in Q$ will still be used to indicate a generic point in the environment. We denote as $\mathcal{P} = \{p_1, \ldots, p_N\} \subset \mathbb{R}^2$ the set of points corresponding to the positions of the UAVs in the team. Similarly, $\mathcal{O} = \{o_1, \ldots, o_{N_{\text{OBS}}}\} \subset \mathbb{R}^2$ indicates the positions of $N_{\text{OBS}} \in \mathbb{N}$ obstacles.

For the sake of simplicity, following [71], we assume the UAVs move slowly enough to approximate their planar motion with a single-integrator model, so Equation (2.2) is still consistent. In this Section, the state is augmented to include the yaw angle in order to explicitly account for sensing and field-of-view constraints, while preserving the single-integrator abstraction for translational motion; we have $p_i = [p_{x_i}, p_{y_i}, \theta_i]^T$ of the i-th robot, and $u_i \in \mathbb{R}^3$ represents still the vector of first-order input control, encompassing both linear and angular velocities. We assume the team is split into $C \in \mathbb{N}$ clusters of $n \in \mathbb{N}$ UAVs, and each UAV is statically assigned to a specific cluster from the beginning of the task execution. For simplicity, we assume N is a multiple of n, ensuring $C = N/n \in \mathbb{N}$. Then, we denote as $G_k \in \mathbb{R}^2$ the centroid of the k-th cluster, and we define a virtual agent $\psi_k \in \mathbb{R}^2$ to be tracked by G_k, as we will detail in the following Sections.

In this hierarchical formulation, coverage optimization is performed at the cluster level rather than at the level of individual robots. Accordingly, Voronoi partitions and centroidal updates are defined with respect to the virtual agents ψ_k, which act as representatives of each cluster. Individual UAVs contribute to coverage indirectly by maintaining formations that enable effective sensing around their associated virtual agent. Consequently, $\mathscr{P} = \{\psi_1, \ldots, \psi_C\}$ will refer to the set of virtual agents' positions. We consider robots with communication capabilities within a communication range $R_c > 0$, and sensing capabilities by means of an on-board monocular camera. To synthesize; $\mathcal{P} = \{p_1, \ldots, p_N\}$ is the formulation when we refer to agents, $\mathscr{P} = \{\psi_1, \ldots, \psi_C\}$ when

we refer to the virtual agents, so the clusters.

For simplicity, the area on the ground captured by the camera is modelled as a circular sector with radius $R_s > 0$ and aperture $\beta > 0$. However, the use of a monocular camera does not allow features localization on the ground, and here comes the need for a cooperative triangulation among robots in a cluster. Therefore, we consider the sensing region of a cluster as the intersection of the fields of view of the UAVs belonging to that sub-group, as depicted in Figure 4.11. In this region, features can be detected by each single UAV and localized through cooperation among them. We assume robots are equipped with odometry sensors for self-localization, and we will indicate as p_j the position of robot j relative to the common global reference frame, while ${}^{i}p_j$ will refer to the position of j relative to the inertial reference frame of robot i. Finally, we will use the notation p_j^k to indicate that robot j is a member of the k-th cluster.

To avoid ambiguity with previous Chapters, we note that symbols reused in this section may carry a different semantic role. In particular, u_i denotes the control input resulting from the constraint-based optimization, rather than a pure coverage descent direction; p_i refers to the position of a physical UAV within a cluster; and V_k denotes a Voronoi region associated with a virtual agent rather than an individual robot. These distinctions reflect the hierarchical nature of the proposed control architecture.

4.2.3.2 Background

Here, we briefly provide the theoretical background for the main tools employed in our solution. In our work, we make use of Control Barrier Functions (CBFs) to ensure collision avoidance with both obstacles and other robots, in conjunction with Control Lyapunov Functions (CLFs) to bring the UAVs of a sub-group to reach a desired configuration. According to [72], let $\mathcal{C} = \{p_i \in \mathbb{R}^m : h(p_i) \geq 0\}$ be the set of configurations that satisfy the requirements for the system, also known as the safe set $\mathcal{C}$, defined as the super level set of a smooth function $h : \mathbb{R}^m \to \mathbb{R}$ with $\frac{\partial h}{\partial p_i}(p_i) \neq 0, \forall p_i \in \partial\mathcal{C}$. CBF properties enable the derivation of an optimization-based controller, which produces a control input u that guarantees constraint satisfaction with the least amount of alteration to a desired input u^*. The optimization problem is formulated as:

$$u(p) = \underset{u \in \mathbb{R}^3}{\arg\min} \ \frac{1}{2}\|u - u^*\|^2 \qquad \text{s.t. } \dot{h}(p, u) \geq -\alpha(h(p)) \tag{4.9}$$

where α is an extended class $\mathcal{K}$ function. Instead, a CLF is a continuously differentiable, positive definite function $\nu : \mathcal{W} \subset \mathbb{R}^m \to \mathbb{R}^+$. As detailed in [73], CLFs can be exploited to asymptotically stabilize the system to a desired state, imposing:

$$\dot{\nu}(\cdot) \leq -\zeta(\nu(\cdot)) \tag{4.10}$$

where, similarly to the CBFs formulation, ζ is another extended class $\mathcal{K}$ function.

4.2.4 Robots Control Layer

The introduction of clustered UAV sub-groups is motivated by sensing limitations inherent to monocular vision. Since a single monocular camera does not provide depth information, reliable target localization requires multiple viewpoints with sufficient geometric diversity. By organizing UAVs into small formations, persistent multi-view observability is ensured, enabling cooperative triangulation while maintaining continuous coverage of the environment.

The inner control layer aims at controlling UAVs of the same generic k-th cluster to track the virtual agent ψ_k, whose motion will be detailed in the following Section. More in details, UAVs within a cluster are required to keep their virtual agent in the center of their field of view, and arrange in order to have the centroid of the cluster coincident with the virtual agent. To simplify the discussion, in the rest of the paper we consider a constant number of UAVs in a cluster, specifically $n = 3$. This requirements translate into a desired formation to be obtained by cluster k, where UAVs are located on a circumference with radius $R_s/2$, placed at an angular distance of 120° to each other. This configuration allows the UAVs to frame the same area with their cameras from different angles, having a complete coverage of the surface for cooperative detection and localization of targets. In addition, robots are also required to avoid collisions with obstacles and other robots, and to not lose communication with other UAVs of their cluster.

4.2.4.1 CBFs for Safety Constraints

The previously cited requirements are met through the utilization of the earlier mentioned CBFs and CLFs. Firstly, we define, for each drone i, the following $h(\cdot)$ functions for every neighbour $j \neq i$ in the k-th cluster:

$$\begin{bmatrix} h_1({}^i p_j^k) \\ h_2({}^i p_j^k) \end{bmatrix} = \begin{bmatrix} \|{}^i p_j^k\|^2 - D_s^2 \\ -\|{}^i p_j^k\|^2 + R_c^2 \end{bmatrix}. \tag{4.11}$$

The first component expresses the difference between the distance from robot j to i and a safety distance $D_s \in R^+$, the second one instead compares the distance between the two drones with the communication range R_c. Imposing $h(\cdot) \geq 0$ constrains the distance between i and j to be greater than a safety threshold, but lower than the communication range in order to maintain communication with each other. Similarly, we define two additional components to ensure collision avoidance with robots from other clusters $c \neq k$ and every obstacle $o \in \mathcal{O}$:

$$\begin{bmatrix} h_3({}^i p_j^c) \\ h_4({}^i o_j) \end{bmatrix} = \begin{bmatrix} \|{}^i p_j^c\|^2 - D_s^2 \\ \|{}^i o_j\|^2 - D_s^2 \end{bmatrix}. \tag{4.12}$$

Then, we define the class $\mathcal{K}$ function $\alpha(\cdot)$ as:

$$\alpha(\cdot) = \gamma h(\cdot) \tag{4.13}$$

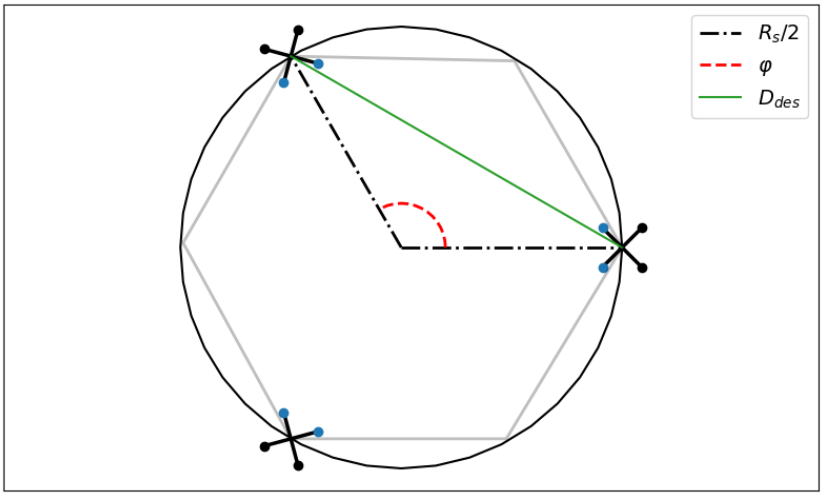


Figure 4.12: Desired distance D_{des} among robots, calculated as the chord linking two points with angular distance φ located on a circumference with radius $R_s/2$.

where $\gamma \in \mathbb{R}^+$ is a positive constant. The time derivative can be expressed as:

$$\dot{h}({}^i p_j) = \frac{\partial h({}^i p_j)}{\partial {}^i p_j} {}^i\dot{p}_j = \begin{bmatrix} -2{}^i x_j & -2{}^i y_j \end{bmatrix} {}^i\dot{p}_j = A_c{}^i\dot{p}_j. \tag{4.14}$$

Since we assumed the UAVs move slowly in the environment, we approximate the relative motion of robot j with respect to i as negligible when enforcing inter-agent constraints. For static obstacles, the obstacle position is constant in the world frame; however, the corresponding barrier function still varies due to the motion of the UAV itself. Thus, we can reformulate Equation (4.14) as:

$$\dot{h}({}^i p_j) \approx A_c{}^i u_i. \tag{4.15}$$

Finally, we can write the optimization problem (4.9) as:

$$\begin{aligned} \arg\min_{u_i \in \mathbb{R}^3} \ & \frac{1}{2}\|u_i - u^*\|^2 \\ \text{s.t. } & A_c{}^i u_i \geq -\gamma h(\cdot) \end{aligned} \tag{4.16}$$

4.2.4.2 CLFs for Desired Configuration

After defining the set of safety constraints robots have to satisfy in order to avoid collisions with obstacles and other robots and to preserve communication with neighbors, we define how the system is controlled to reach the desired configuration. This requirement is rephrased as an objective distance that each member of the k-th cluster should maintain from one another: $D_{\text{des}} \in \mathbb{R}^+$ is equivalent to the chord linking two points on a circumference with radius $R_s/2$, with angular distance $\varphi = 120°$:

$$D_{\text{des}} = 2\frac{R_s}{2}\sin\frac{\varphi}{2} = R_s \sin\frac{\varphi}{2} \tag{4.17}$$

as can be inferred from the representation in Figure 4.12.

Then, we can define the CLF as:

$$\nu({}^i p_j^k) = (\|{}^i p_j^k\| - D_{des})^2. \tag{4.18}$$

It is easy to see that ν is positive definite and continuously differentiable function, thus a suitable candidate CLF.

With the same considerations that have been made for the CBFs, we can write:

$$\dot{\nu}({}^i p_j^k) = \frac{\partial \nu({}^i p_j^k)}{\partial {}^i p_j^k} {}^i \dot{p}_j^k \approx {B_c}^i u_i. \tag{4.19}$$

Stabilization to the desired state is carried out defining the extended class $\mathcal{K}$ function $\zeta(\nu(\cdot))$ as:

$$\zeta(\nu(\cdot)) = \kappa\nu(\cdot) \tag{4.20}$$

where $\kappa \in \mathbb{R}^+$ is a positive constant, and integrating the constraint (4.10) into the optimization problem (4.16).

The problem then evolves into:

$$\begin{aligned} \underset{u_i \in \mathbb{R}^3}{\arg\min}\ & \frac{1}{2}\|u_i - u^*\|^2 + \varepsilon \\ \text{s.t. } & {A_c}^i u_i \geq -\gamma h(\cdot) \\ & {B_c}^i u_i \leq \kappa\nu(\cdot) + \varepsilon \end{aligned} \tag{4.21}$$

It should be noted that a slack[1] variable $\varepsilon \in \mathbb{R}^+$ has been introduced into the constraint equation; this addition aims to prioritize the adherence to safety constraints over achieving the desired configuration. Thus, UAVs within a cluster will strive to maintain the desired distance from each other, but deviations from the desired configuration will be allowed to avoid collisions with obstacles and other robots. Finally, what is left is the definition of the desired control input u^*. As previously mentioned, the desired velocity makes each robot move in such a way that the centroid of the k-th cluster tracks a virtual agent ψ_k, whose motion will be detailed in the following section. Let $G_k \in \mathbb{R}^2$ be the centroid of cluster k, whose position can be calculated independently by each UAV within the sub-group, thanks to the always available communication channel ensured by CBFs. Approximating the cluster to a rigid body, we can calculate the linear velocity of each UAV as the linear velocity required by G_k to reach ψ_k. The angular velocity instead, is calculated to keep ψ_k in the middle of the field of view. Hence, the desired velocity can

[1] A slack variable is an auxiliary non-negative decision variable introduced to relax a constraint, allowing controlled constraint violation when strict feasibility cannot be guaranteed. In this context, it enables safety constraints to be satisfied with higher priority than formation objectives by permitting bounded deviations from the desired configuration.

be formulated as follows, relative to the inertial reference frame of robot i:

$$ {}^{i}u_i^* = K_p \begin{bmatrix} {}^{i}\psi_x^k - {}^{i}G_x^k \\ {}^{i}\psi_y^k - {}^{i}G_y^k \\ \text{atan2}\big({}^{i}\psi_y^k - {}^{i}G_y^k,\ {}^{i}\psi_x^k - {}^{i}G_x^k\big) \end{bmatrix} \tag{4.22} $$

where $K_p \in \mathbb{R}^+$ is a proportional gain.

4.2.5 Clusters Coordination Layer

As previously mentioned, the overall control strategy can be seen as a two-level approach, where the lower level aims at coordinating UAVs within a cluster, and the higher control coordinates clusters with each other. After presenting the lower control level, we detail in this section the higher one, responsible for moving the virtual agent ψ_k. The method being presented is semi-centralized, meaning that only one member of the cluster k calculates and communicates ψ_k to the other members. It is important to note that every robot is able to carry out the virtual agent's control, but a centralized solution ensures accordance among members of the same cluster, avoiding mismatches in calculating ψ_k in a decentralized manner. In addition, a communication link is ensured to exist among robots within a cluster thanks to the CBFs constraints defined in the previous section.
Consistent with the Voronoi-based coverage framework introduced in Chapter 2, Lloyd-type descent dynamics are applied to the virtual agents to drive the system toward a centroidal Voronoi configuration at the cluster level, coverage control brings the system to the configuration that maximizes the amount of information the swarm can gather, as indicated by a probability density function $\phi(q) : \mathbb{R}^2 \to \mathbb{R}^+$. In our work, we exploit the Voronoi algorithm to generate a partitioning of the environment from virtual agents' positions $\psi \in \mathscr{P}$. Since UAVs do not have global knowledge, a limited-range decentralized implementation [47] of the Voronoi partitioning is carried out. The resulting generic Voronoi cell $V_k \subset Q$ includes zones of the environment that are closer to the k-th virtual agent than to any others, and these zones fall within the sensing area of ψ_k. Given that UAVs asymptotically move to the configuration detailed in the previous section, where they are located on a circumference centered in ψ_k with radius $R_s/2$, we approximate the virtual sensing region of a generic cluster as the closed ball $B(\psi_k)$ centered in ψ_k with radius $R_s/2$. The formulation for the Voronoi cell then becomes:

$$ V_k = \{q \in B(\psi_k) | \|q - \psi_k\| \leq \|q - \psi_j\|, \forall \psi_j \in \mathscr{P} \setminus \{\psi_k\}\} \tag{4.23} $$

Assuming that robots from different clusters can exchange the position of their virtual agents, the limited Voronoi cell V_k can be constructed from the locally available information through communication. Then, following traditional coverage control, the centroid of the region is calculated taking into account the probability density function $\phi(q)$ of the environment:

$$ C_{V_k} = \frac{\int_{V_k} \phi(q) q dq}{\int_{V_k} \phi(q) dq}. \tag{4.24} $$

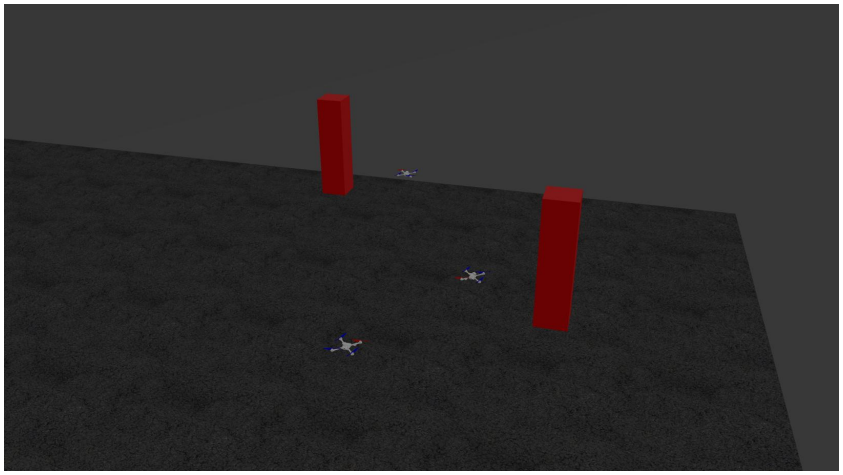


Figure 4.13: Gazebo virtual environment with 3 UAVs and 2 obstacles.

Finally, the control input for ψ_k follows the proportional law:

$$u_{\psi_k} = K_p(C_{V_k} - \psi_k), \tag{4.25}$$

moving iteratively ψ_k towards the centroid of its Voronoi region, and eventually moving the system to reach a centroidal Voronoi configuration, which optimizes the total area covered. The virtual position of ψ_k is constantly shared with other members of cluster k, allowing each of them to calculate its own desired control input with (4.22).

4.2.6 Experimental Evaluation

The experiments presented in this section are designed to validate the constraint-aware execution layer introduced in this chapter, complementing the coverage performance analysis of Section 4.1; we present the experimental evaluation of our two-level control strategy. The tests were conducted in two phases: first, we evaluated the behavior of a single cluster to ensure that the designed CBFs and CLFs achieved the desired configuration and avoided collisions. In the second phase, we assessed the overall performance of the entire team and the coordination among clusters, measuring the coverage performance of the multi-robot system.

4.2.6.1 Single-Cluster Tests

In the first set of tests, we controlled a single cluster composed by 3 UAVs to cooperatively track a virtual agent while reaching a desired formation and avoiding obstacles in the environment. The virtual simulation environment was created using Gazebo and virtual UAVs from the RotorS simulation framework [74], controlled with the ROS middleware [75]. Obstacles were created within the virtual environment (see Figure 4.13) and randomly placed in different locations for each simulation run. UAVs were also randomly initialized in a location inside the safe set $\mathcal{C}$, which means the distance between one another is greater than the safe distance and lower than the communication range, satisfying CBFs constraints. The safety distance has been set to $D_s = 2.0$ m, commu-

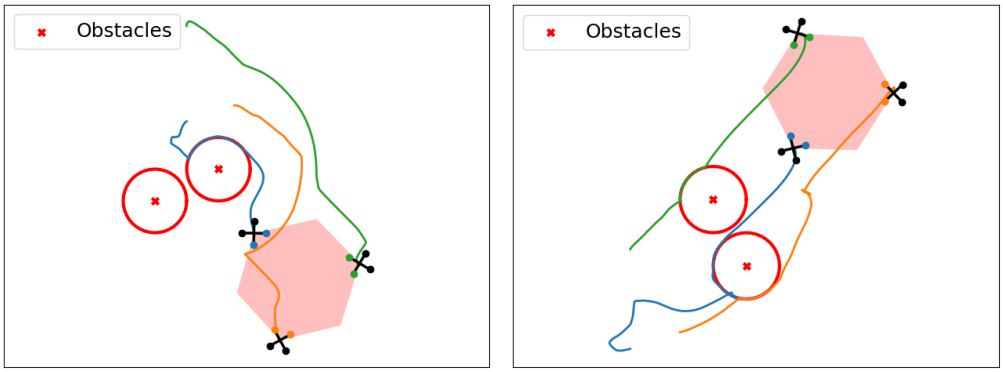


Figure 4.14: Trajectories of 3 UAVs required to reach a desired formation while always keeping a safety distance from obstacles.

nication range to $R_c = 10.0$ m, and the field of view as an circular sector with radius $R_s = 10.0$ m and amplitude $\beta = 120°$. Parameters in the optimization problem (4.21) were set to $\gamma = 5.0$ and $\kappa = 0.1$ to prioritize safety rather than formation keeping, and the slack variable was set to $\varepsilon = 1.0$.

Robots in this first experimental phase were required to provide coverage in an environment whose probability density $\phi(q)$ was simply described as a bivariate Gaussian distribution, defined by a mean point $\mu \in \mathbb{R}^2$, randomly generated for each run, and a covariance matrix $\Sigma \in \mathbb{R}^{2\times2}$, which was assumed to be constantly equal to the identity matrix. Controlling just one cluster and, consequently, one virtual agent, results in the virtual agent being moved to the mean point μ and does not offer any insightful information about the effectiveness of the overall coverage strategy. Nonetheless, this initial test phase demonstrated the UAVs' ability to achieve the desired configuration while maintaining a safety distance from obstacles in the environment.

As shown in Figure 4.14, robots always try to keep the desired configuration while moving, only modifying their configuration when safety requirements necessitate avoidance of obstacles. Particularly interesting is also the analysis of the CBFs and CLFs values during the execution of the task, shown in Figure 4.15. Here, constraint equations of the optimization problem (4.21) have been reformulated such that $\dot{h}(\cdot) + \gamma h(\cdot) \geq 0$ and $-\dot{\nu}(\cdot) + \kappa\nu(\cdot) \geq -\varepsilon$. For clarity, only the effect of CBFs for obstacle avoidance is shown, together with the corresponding values for the CLFs. In particular, the upper plot shows how the CBF constraint is always satisfied, and the curve does never reach negative values. In the meantime, the CLFs curves shown in the lower plot stay close to 0 for the majority of time, but they are allowed to assume negative values when the robot is required to avoid an obstacle, reflecting the cluster's tendency to stray from the intended configuration when necessary for safety reasons.

4.2.6.2 Multi-Cluster Coordination

After assessing performances of the inter-agent coordination and collision avoidance within a single cluster, we conducted a second set of tests to evaluate coordination among different clusters and the overall performances of the multi-robot system. The simulation

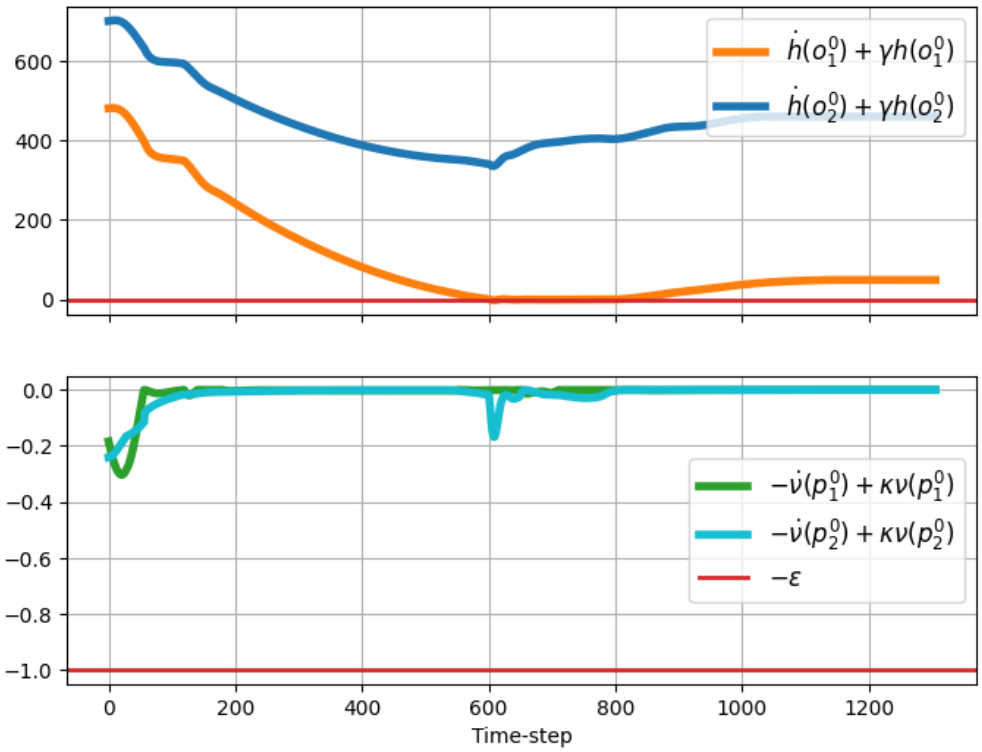


Figure 4.15: Values of the CBFs and CLFs during task execution.

setup has been the same as the previous tests, with the only difference of defining more complex probability densities, modelling $\phi(q)$ as the sum of different bivariate Gaussian distributions, or a Gaussian Mixture Model. A leader UAV was chosen among each cluster, this is in charge of defining the motion of the virtual agent and communicate its position to its neighbors. Communication was allowed within a communication range $R_c = 10.0$ m, agents from different clusters $j \neq k$ share information about their respective virtual agent, while agents of the same cluster k share information gathered from other robots, other than the position of ψ_k.

Several simulation runs have been carried out and the configuration adopted by the cluster was evaluated; Figure 4.16 shows the trajectories and the final configuration of every robot in different scenarios. In Figure 4.16a and Figure 4.16b, $\phi(q)$ is defined as a Gaussian Mixture Model, highlighting continuous regions of the environment where the likelihood of finding events of interest is higher. In these scenarios, the UAVs move in order to cooperatively monitor the area of interest, and their final position results as a balancing between the collective task to be accomplished with other members of the same cluster, and the requirement of keeping a safety distance from all the other UAVs. Differently, the likelihood density $\phi(q)$ in Figure 4.16c results from 4 different Gaussian components. Since the number of Gaussian components is equal to the number of clusters, each cluster ends up monitoring one of them. Finally, an interesting behavior emerges from Figure 4.16d. Here, $\phi(q)$ is defined as a single bivariate Gaussian distribution, hence all 3 UAV clusters will be attracted into the same area. As we can see, robots arrange themselves forming a single global cluster, where all the robots redundantly face the same region. This behavior would be even more emphasised for a more concentrated distribution, with the final overall configuration better approximating a circumference,

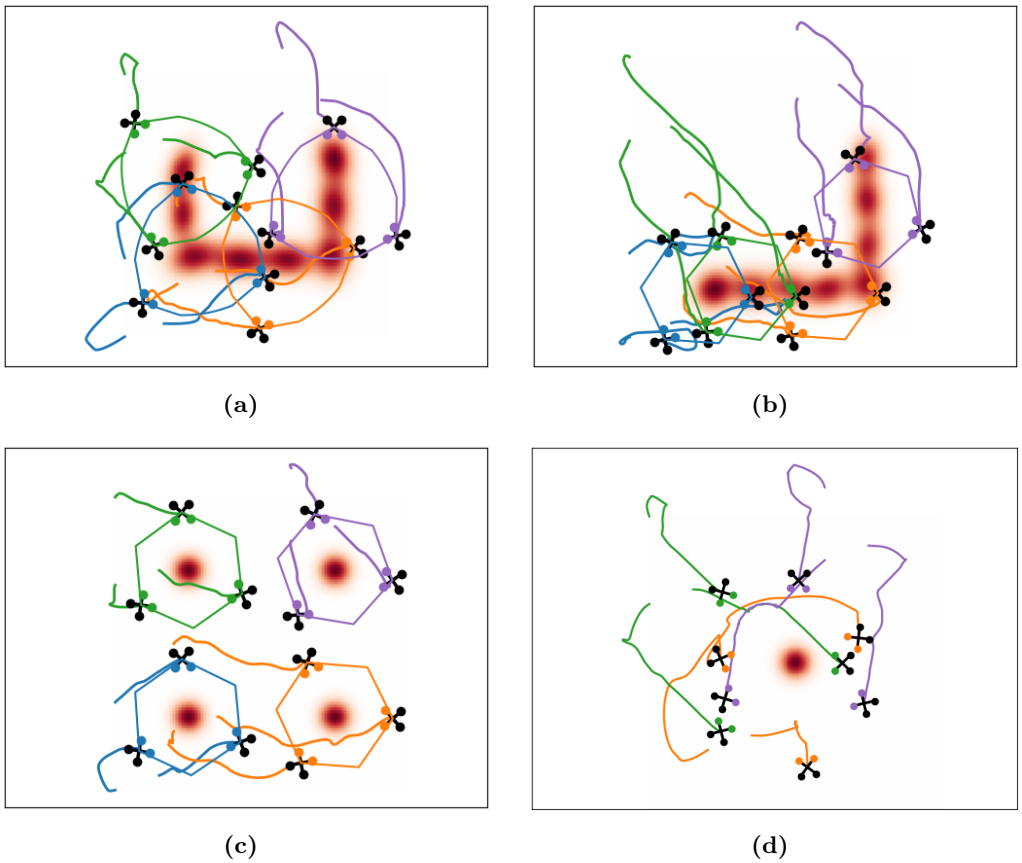


Figure 4.16: Results in different scenarios, where $\phi(q)$ is defined as a Gaussian Mixture Model (a, b), sum of 4 Gaussian distributions (c), and a single Gaussian function (d).

reflecting a greater degree of certainty regarding the location of events of interest and therefore arranging around it.

Figure 4.17 shows a time frame of a real drones experiment, the two set of drones—in blue and red—execute coordinated motions in close proximity. While the virtual targets induce intersecting trajectories that would lead to collisions if followed exactly, the safety layer actively modifies the commanded velocities to guarantee separation constraints. As illustrated by the overlapping paths and directional markers, collision avoidance interventions occur locally and transiently, without disrupting the global formation objective of either subgroup.

4.2.7 Discussion

The results presented in this Section demonstrate how coverage control objectives can be executed reliably in the presence of safety, sensing, and coordination constraints by embedding them within a constraint-aware control architecture. Rather than modifying the underlying Voronoi-based coverage formulation, the proposed approach augments it with a hierarchical execution layer that ensures feasibility on physical platforms. This separation between coverage optimization and constraint handling proves effective in preserving the desirable properties of Lloyd-based coordination while accommodating the non-idealities inherent to real multi-UAV systems.

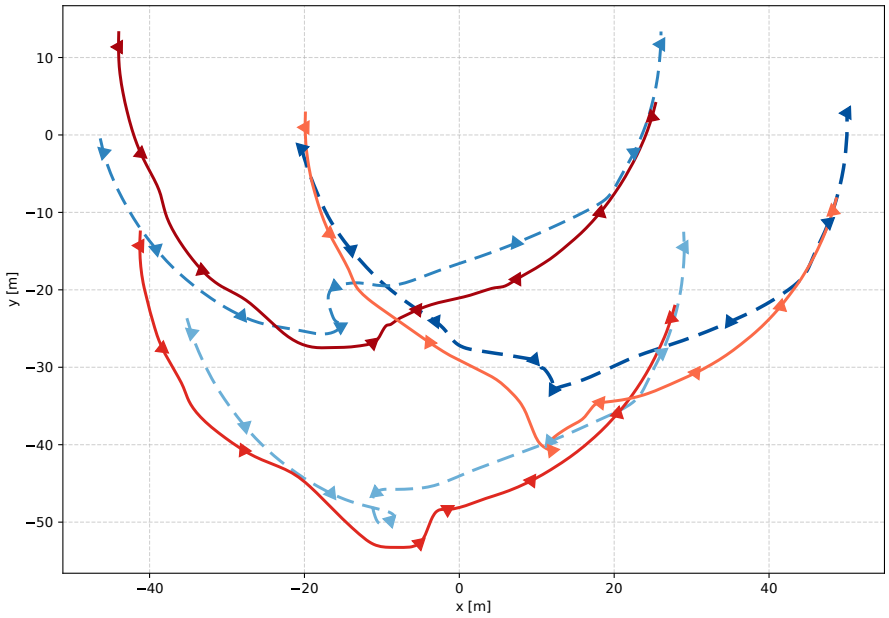


Figure 4.17: Planar trajectories of two coordinated UAV subgroups over a selected time window. The dashed blue curves and solid red curves represent two distinct subgroups assigned to two different virtual targets moving along the same circumference in counter-phase. As the virtual targets evolve, the two formations approach each other and eventually meet along overlapping portions of the trajectory. Throughout the maneuver, each subgroup preserves its internal formation through the CLF-based control action. When the inter-group distance becomes too small, the CBF-based safety layer modifies the nominal motion to guarantee collision avoidance. Once the subgroups are again sufficiently separated, the UAVs return to their nominal formation around the corresponding virtual targets. The figure therefore illustrates the collision-avoidance capability of the proposed framework while maintaining coordinated formation behavior.

The experimental results highlight the role of Control Barrier Functions in mediating conflicts between coverage-driven motion commands and safety requirements. When virtual agent trajectories induce intersecting or potentially colliding paths, the safety layer locally modifies the commanded velocities to enforce separation constraints, without disrupting the global coordination objective. These interventions are transient and localized, allowing the system to recover the desired formation once the unsafe condition has been resolved. This behavior confirms that safety constraints can be enforced without introducing long-term degradation in coordination performance.

The use of clustered formations further illustrates how sensing limitations influence control design. By assigning coverage objectives to virtual agents and constraining physical UAVs to maintain formations around them, the framework decouples perception requirements from coverage optimization. This abstraction enables cooperative triangulation with monocular sensors while maintaining compatibility with distributed coverage control. The resulting behavior, observed across multiple density configurations, reflects a balance between sensing-driven formation maintenance and coverage-driven spatial allocation.

At the same time, the experiments also expose intrinsic limitations of the proposed formulation. When multiple clusters are attracted toward highly concentrated density regions, the system naturally converges to configurations where sensing redundancy increases. While this behavior is consistent with the coverage objective, it highlights the absence of

explicit mechanisms to penalize redundant sensing or promote cluster dispersion beyond safety constraints. Such effects are not failures of the control architecture, but rather consequences of the chosen coverage metric and abstraction level.

4.3 Principal Results and Observations

This Chapter examined distributed coverage control under time-varying and adaptive conditions, with the objective of assessing the practical limits of Voronoi-based methodologies when deployed on realistic multi-UAV systems. Through a combination of algorithmic analysis, simulation studies, and real-world experiments, several key findings emerge.

First, Voronoi-based distributed coverage remains effective under time-varying spatial conditions, provided that the rate of environmental change is compatible with the agents' motion and sensing capabilities. When points of interest evolve continuously, the limited-range decentralized controller is able to track centroidal configurations with bounded degradation in coverage quality. This confirms that classical coverage formulations, originally developed for static environments, can be extended to dynamic settings without requiring global coordination or centralized re-planning.

Second, the quality of coverage in time-varying scenarios is strongly influenced by the realization of the coverage control law, rather than by the Voronoi partitioning itself. Experimental results show that standard proportional Lloyd-based controllers are prone to oscillations and overshoot when deployed on real aerial platforms. Introducing smooth, state-dependent control gains significantly improves stability and convergence behavior, enabling reliable operation in the presence of sensor noise, actuation delays, and communication imperfections.

Third, distributed coverage controllers exhibit robustness to changes in swarm size and composition, including agent addition and removal, as long as local sensing and communication assumptions are preserved. The Voronoi framework naturally adapts to these changes through local partition updates, allowing the system to reconfigure without explicit coordination mechanisms or global state knowledge.

Fourth, coverage control can be successfully embedded within safety- and task-constrained control architectures by decoupling coverage optimization from constraint enforcement. The hierarchical framework presented in Section 4.2 demonstrates that coverage objectives can coexist with collision avoidance, formation maintenance, and environmental constraints when enforced through control-theoretic mechanisms such as control barrier functions. While safety constraints may locally override optimal coverage behavior, the system preserves global task coherence and remains stable.

Finally, experimental validation confirms the practical feasibility of adaptive distributed coverage control for multi-UAV systems. Field trials with quadrotor teams demonstrate that the proposed approaches are not limited to simulation environments, but can be deployed on real platforms operating under realistic sensing, communication, and actuation

constraints. The observed behaviors closely match theoretical expectations, reinforcing the relevance of Voronoi-based coverage as a scalable coordination primitive for aerial robotic swarms.

Chapter 5

Multi-Dimensional Coverage Control

Multi-robot coverage control has been extensively studied in planar domains, where classical geometric methods—such as Voronoi-based partitions and Lloyd-type algorithms—provide strong guarantees on convergence, decentralization, and computational efficiency. However, many real-world applications involving aerial and heterogeneous robot teams naturally arise in three-dimensional environments or on curved manifolds embedded in $\mathbb{R}^3$. Examples include UAV inspection of large infrastructures, cooperative perception around a subject, surface inspection of aircraft, and volumetric environmental monitoring. In such settings, both the geometry of the workspace and the combinatorial structure of the underlying partitioning methods differ fundamentally from the planar case.

Figure 5.1 illustrates a representative scenario in which a swarm of drones cooperatively inspects an aircraft fuselage [76]. Such tasks require viewpoint diversity, collision-free coordination, and real-time adaptation over a curved three-dimensional surface, highlighting the need for coverage techniques beyond standard planar formulations.

Figure 5.1: Example of multi-UAV infrastructure inspection: Korean Air's autonomous drone swarm performing coordinated aircraft-fuselage inspection.

This chapter addresses coverage control in such multi-dimensional domains. Section 5.1

reviews coverage formulations and Voronoi structures beyond the plane, emphasizing how higher-dimensional geometry and combinatorial complexity affect algorithm design. Section 5.2 introduces hemispherical viewpoint coverage via a sweep-based spherical Voronoi construction and density-aware centroidal control law. Section 5.3 presents an applied implementation of hemispherical coverage control validated in multi-drone simulation and field experiments. Finally, Section 5.4 concludes the Chapter with discussions and analysis.

5.1 Coverage Control in Higher Dimensions

Many multi-robot coordination problems require deployment over domains that cannot be approximated as planar regions. Coverage tasks arise in volumetric environments ($Q \subset \mathbb{R}^3$), on curved surfaces such as aircraft fuselages or industrial structures, and around targets where viewpoint diversity is required. Extending classical planar coverage methods to these settings is non-trivial: Voronoi cells become polyhedra or surface patches, distance metrics must account for curvature, and the computational complexity of partitioning grows significantly with dimension. This Section formalizes the higher-dimensional coverage problem and reviews representative approaches for volumetric and surface-based coverage, laying the foundation for the hemispherical framework presented in the following sections.

5.1.1 Problem Statement

Let $Q \subset \mathbb{R}^d$ with $d \geq 2$ denote the domain of interest, and let $\mathcal{P} = \{p_1, \ldots, p_n\}$ be the positions of n agents. As in the planar case, a standard objective is to optimize the density-weighted performance functional:

$$\mathcal{H}(\mathcal{P}) = -\sum_{i=1}^{n} \int_{V_i(\mathcal{P})} \|q - p_i\|^2 \, \phi(q) \, dq, \tag{5.1}$$

where $V_i(\mathcal{P})$ denotes the Voronoi region assigned to generator p_i under an appropriate metric. When $d = 2$, Voronoi cells are polygons, centroid computation is straightforward, and Fortune's algorithm allows Voronoi updates in $O(n \log n)$ time, given Lloyd's methodology.

In higher dimensions, however, Voronoi cells become polyhedra or curved surface patches, and both their construction and centroid evaluation become significantly more complex. Boissonnat *et al.* [77] show that the combinatorial complexity of Voronoi diagrams under polyhedral distance metrics grows as[1]:

$$\Theta\left(n^{\lceil d/2 \rceil}\right),$$

which for $d = 3$ can reach quadratic complexity even under relatively simple metrics.

[1] $\Theta(\cdot)$ indicates a tight asymptotic bound on the combinatorial complexity of the diagram, rather than an upper bound on algorithmic runtime as it is for $O(\cdot)$.

These results illustrate why naively "lifting" a 2D Voronoi-based coverage algorithm to $\mathbb{R}^3$ leads to substantially higher computational cost, and why specialized algorithmic treatments are required.

5.1.2 From Planar to Multi-Dimensional: Volumes and Surfaces

5.1.2.1 Voronoi-Based Coverage in Full 3D

A representative example of full volumetric coverage is the 3D-Voronoi Partition Coverage Algorithm (3D-VPCA) proposed by Dang *et al.* [78]. The method considers a wireless sensor network deployed in a three-dimensional region, where each node must position itself to maximize target detection while respecting sensing constraints. The workspace is decomposed into a *3D Voronoi tessellation* (the "V-body" partition), and nodes move under virtual forces to maximize coverage quality.

The procedure begins with a random initialization of nodes in the 3D domain, followed by repeated evaluation of neighbor relations, force computation, and node displacement. The core computational step involves the aggregation of interaction forces between each node and all other nearby nodes, which leads to a per-iteration cost of $O(n^2)$—dominating the one-time cost of constructing the initial 3D Voronoi tessellation. The Algorithm 1 is reported for reference.

Algorithm 1 3D-VPCA: 3D Voronoi Partition–Coverage Algorithm [78]

Require: Number of sensor nodes n, sensing radius R_s, workspace bounds $(x_{\max}, y_{\max}, z_{\max})$

Ensure: Final node positions and network coverage $S(A)$

Initialization:
  Divide workspace into 3D Voronoi "V-bodies" v_i
  $N_{\text{iter}} \leftarrow 100$, max_step $\leftarrow [0, 10]$
Randomly initialize node positions $S_i = (A_i, B_i, C_i)$
Compute initial 3D Voronoi cells $\{v_1, \ldots, v_n\}$
for $i = 1$ to n **do**
  if $d_{ij} \leq R_s$ **then**
    Save current node s_k and Voronoi unit v_k
  else
    Select free node $s_f \leftarrow s_i - s_k$
    if $d_{ij} > R_s$ **and** $N_{c_j} \neq 0$ **then**
      $F_A = \sum_{j \neq i} F_{ij} + F_a + F_c$
      Move s_f toward neighbor cell N_{c_i}
    else
      Move s_f toward boundary node s_L
    end if
  end if
end for
return Updated nodes $\{S_i\}$ and coverage $S(A)$

Figure 5.2 illustrates the type of 3D Voronoi "V-body" decomposition used in [78].

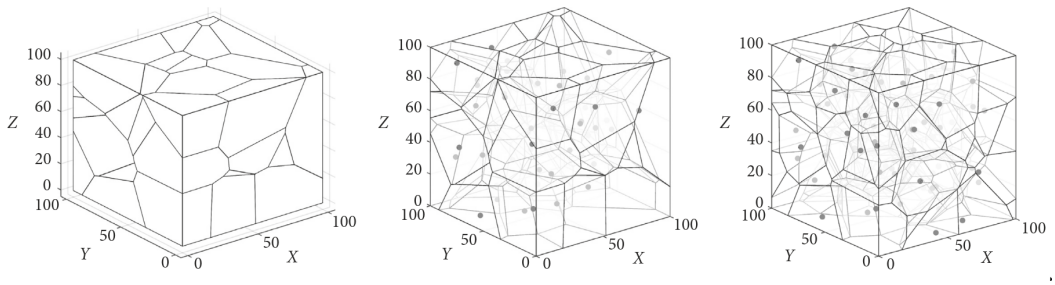


Figure 5.2: Example of a 3D Voronoi ("V-body") decomposition used in Dang *et al.*

5.1.2.2 Surface-based 3D Coverage

A different approach to higher-dimensional coverage is presented by Breitenmoser *et al.* [79], who consider robots deployed on a surface embedded in $\mathbb{R}^3$ rather than a full 3D volume. In this case, the domain is a two-dimensional manifold (represented as a triangle mesh), and the authors construct Voronoi partitions directly on the mesh.

Although the environment lives in 3D, the coverage computation effectively becomes a *surrogate planar problem*. Regions are defined along the mesh graph, and the Lloyd's algorithm is run in this discrete planarized domain. The centroid of each cell is computed using mesh geometry, but no volumetric computation is required.

This allows the authors to reuse classical planar Voronoi coverage techniques while only handling the 3D geometry during projection and mesh traversal. As such, this approach is powerful when the surface admits a well-behaved mesh representation, but it cannot be directly used for volumetric coverage or for manifold domains with complex topology or discontinuities.

Figure 5.3 illustrates how the surface is discretized into a triangular mesh and how planar-like Voronoi partitions are computed on top of it [79].

5.1.2.3 Summary

The two approaches reviewed above illustrate complementary strategies for extending coverage control to higher-dimensional domains. Fully three-dimensional methods, such as the 3D-VPCA algorithm of Dang *et al.*, operate directly on volumetric Voronoi polyhedra. While this provides an exact representation of spatial responsibility regions, it comes with a significant computational burden: force-based or neighbor-dependent updates typically incur at least $O(n^2)$ complexity per iteration, and the underlying 3D Voronoi construction itself may scale as $\Theta(n^2)$ in the worst case. Such costs quickly become prohibitive when the number of agents increases.

Conversely, surface-based methods—exemplified by the work of Breitenmoser *et al.*—avoid volumetric computations altogether by restricting coverage to a two-dimensional manifold represented by a mesh. In this case, the problem is effectively reduced to a quasi-planar

Figure 5.3: Voronoi-based coverage on a surface mesh from Breitenmoser *et al.* Although the surface is embedded in 3D, the coverage algorithm operates on a 2D discretization of the mesh, effectively reducing the problem to a planar surrogate.

setting: Voronoi regions are computed on the graph structure of the surface, enabling the reuse of classical 2D coverage techniques. This reduction maintains algorithmic tractability but limits applicability to domains that can be faithfully approximated as embedded surfaces with consistent parametrizations. It cannot be used for true volumetric coverage, nor for domains whose geometry is not easily discretized into a planar surrogate.

These contrasting limitations motivate the search for a middle ground: a coverage strategy that preserves the low computational complexity of planar methods while handling intrinsically three-dimensional geometry. In this thesis, we adopt the hemispherical viewpoint-coverage domain as such an intermediate case. By operating directly on the sphere (or hemisphere) and redefining distance metrics and Voronoi constructions accordingly, we retain an $O(n \log n)$ complexity similar to 2D Fortune-like methods, while capturing the essential 3D structure required for aerial multi-robot coverage tasks. This enables scalable, distributed coverage on a curved 3D manifold without incurring the high overhead associated with full 3D volumetric approaches.

Although both approaches consider robot motion constrained to a surface embedded in $\mathbb{R}^3$, the underlying problem classes are different. Breitenmoser *et al.* address coverage over a generic triangulated surface, where the mesh acts as a discrete substrate allowing the reuse of planar Voronoi ideas on an embedded manifold. By contrast, the present thesis considers a hemispherical viewpoint-coverage domain with known analytic geometry, where distance, partitioning, and centroidal conditions are defined directly with respect to the spherical manifold. Hence, the contribution here is not simply to perform coverage

on a 3D surface, but to develop a computationally efficient intrinsic formulation tailored to hemispherical geometry, avoiding both full volumetric computation and mesh-based planarization.

5.1.3 Coverage as a Hemispherical Problem: Motivation

In many multi UAV applications, the notion of coverage departs from the classical objective of uniformly tiling a planar region or filling a three-dimensional volume. Instead, agents are required to distribute themselves around a subject of interest so as to maximize *viewpoint diversity*, ensure persistent visibility, and maintain safe relative distances. This class of problems naturally arises in scenarios where the quality of coverage is determined by angular separation and relative orientation rather than by spatial density in Euclidean space [80, 81].

Several works have addressed viewpoint-oriented coordination in multi-UAV systems by introducing parameter spaces for camera motion and cinematographic constraints [81], or by studying formation control strategies tailored to aerial filming tasks [82]. Other approaches rely on greedy coordination schemes or geometric control based on Voronoi to distribute agents around a subject [83, 84]. While effective in specific scenarios, these methods typically operate on planar proxies or local heuristics and often do not explicitly address scalability or computational complexity, which becomes critical as the number of agents increases.

Formulating coverage directly on the hemispherical manifold also enables a unified treatment of the formation geometry and coverage quality. Spatial preferences or user-defined priorities can be incorporated through density functions defined on viewing directions, allowing agents to bias their distribution toward regions of interest while maintaining global coverage. This perspective is consistent with distributed coverage formulations that rely on local sensing and limited communication as in the previously mentioned planar case [2, 85], but extends them to intrinsically three-dimensional, non-planar domains.

5.2 Hemispherical Coverage Control

While Section 5.1 reviewed existing approaches to coverage control in higher-dimensional and non-planar domains, this Section focuses on a specific class of three-dimensional coverage problems: *viewpoint coverage on a hemispherical manifold*. This formulation naturally arises in multi-UAV applications where agents are required to maintain angular diversity around a point of interest—such as infrastructure inspection [86] or aerial cinematography [87]—rather than tile a planar or volumetric region.

Modeling admissible viewpoints as a hemisphere centered on the target provides a geometrically consistent abstraction that preserves rotational invariance and avoids the distortions introduced by planar projections. At the same time, it introduces challenges that are not present in classical planar coverage problems: distances must be defined along spherical domains, Voronoi regions become curved surface patches, and centroid

computation must account for the intrinsic geometry of the manifold.

In this Section, we present a decentralized coverage control methodology that operates directly on the hemispherical surface. The proposed approach extends Lloyd-type coverage control to the spherical setting by combining a sweep-based construction of spherical Voronoi partitions with density-aware centroid computation on the manifold. Using a Fortune-inspired sweep-plane algorithm adapted to the sphere, the method preserves the computational complexity $\mathcal{O}(n \log n)$ of the planar Voronoi coverage while remaining compatible with distributed execution based on local-information.

The objective of this section is to formalize the hemispherical coverage problem, describe the underlying Voronoi partitioning and centroidal control mechanisms, and analyze their computational properties. Implementation aspects, system integration, and experimental validation on real multi-UAV platforms are deliberately deferred to Section 5.3, allowing the present section to focus exclusively on the algorithmic and geometric foundations of hemispherical coverage control.

5.2.1 Spherical Voronoi Diagrams

In this Section, we review the notation and fundamental concepts related to Voronoi diagrams and Fortune's algorithm, together with the necessary considerations arising from the transition of the problem domain from the planar case to the spherical one. Parts of the hemispherical coverage framework presented in this Chapter have been previously introduced in [5].

5.2.1.1 Geodesic Distances

Distances between points on the surface are measured along spherical geodesics, defined by

$$d(p, q) = \arccos(p \cdot q), \tag{5.2}$$

which is the natural metric on the sphere.

5.2.1.2 Voronoi Cell Formulation on the Sphere

In this part we extend the formulation introduced in Section 2.1, we use $\mathbb{F}(\mathbb{R}^3)$ to refer to the family of all finite point sets in $\mathbb{R}^3$. An element $\mathcal{P} \in \mathbb{F}(\mathbb{R}^3)$ can be written as $\{p_1, \ldots, p_n\} \subset \mathbb{R}^3$, where $\{p_1, \ldots, p_n\}$ are individual points. A Voronoi partition can be defined over a polyhedral subset of $\mathbb{R}^3$. In this Section, we denote by $\mathcal{Q} \subset \mathbb{R}^3$ a convex polyhedron representing the workspace to be covered by the robotic agents. Since any sphere can be scaled and shifted, we consider only the tessellation of the unit sphere centered at the origin; we denote the unit sphere as $S^2 = \{\, p \in \mathbb{R}^3 : \|p\|_2 = 1\}$. We consider a set of N distinct points on the sphere, namely $p_1, \ldots, p_N$, indicated as sites. Same as the previous sections, we denote the Voronoi diagram of $\mathcal{P}$ by $\mathcal{V}(\mathcal{P})$, we define $\mathcal{V}_i(p)$, the Voronoi cell for site p_i, to be the set of points on the sphere that are closer to

p_i than to any other site:

$$\mathcal{V}_i(p) = \big\{ \hat{r} \in S^2 : d(\hat{r}, p_i) < d(\hat{r}, p_j), \quad \forall\, j \neq i, \text{ for all } p_i, p_j \in \mathcal{P} \big\}. \tag{5.3}$$

Two agents are said to be Voronoi neighbors if $V_i \cap V_j \neq \emptyset$, where, for brevity, V_i indicates $V_i(\mathcal{P})$. For additional details, the reader is referred to [88]. Figure 5.4 shows an elaboration of a Voronoi partition on the sphere taken from the [89] methodology.

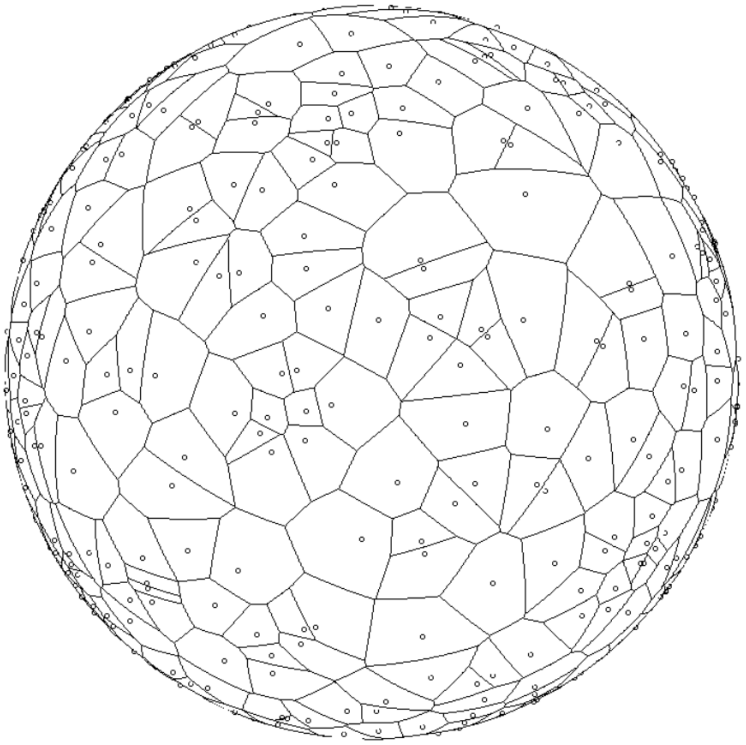


Figure 5.4: Voronoi diagram over a sphere generated from 500 sites

Unlike planar Voronoi diagrams, spherical Voronoi cells are curved surface regions embedded in $\mathbb{R}^3$. Their boundaries are portions of great circles, corresponding to set of points that are equidistant——in the geodesic sense——from two neighboring generators. As a result, each cell V_i is a convex subset of the sphere *in a geodesic definition*, meaning that any shortest geodesic between two points in V_i lies entirely within the cell. This property is fundamental for defining centroidal configurations and for interpreting coverage quality in terms of angular separation on the hemisphere.

The proposed notation can be easily generalized from the spherical case to the hemispherical case by restricting the domain to $(x, y, z) \in \mathbb{R}^3 \mid z > 0$. Restricting the domain to a hemisphere does not alter the definition of the Voronoi cells, but simply truncates the admissible surface.

5.2.2 Fortune's Algorithm for Spherical Diagrams

The construction of spherical Voronoi diagrams adopted in this work builds upon the extension of Fortune's sweep-line algorithm to the surface of the sphere proposed in [89].

As in the planar case, the core idea is to incrementally construct the Voronoi diagram by sweeping a geometric primitive across the domain and handling discrete topological events, while ignoring intermediate configurations.

On the sphere, the sweep line is defined as the intersection between the unit sphere and a moving plane, referred to as the *sweep plane*. As like as in the planar car, the intuition is to incrementally build the Voronoi diagram by maintaining a dynamic data structure known as the beach line and a priority queue of future events, the wo key event types are:

- **Site events**: A site event occurs when the sweep line reaches a new site $\hat{p}_i \in \mathcal{P}$,
- **Circle events**: A circle event occurs when a parabolic arc on the beach line shrinks to zero length and disappears.

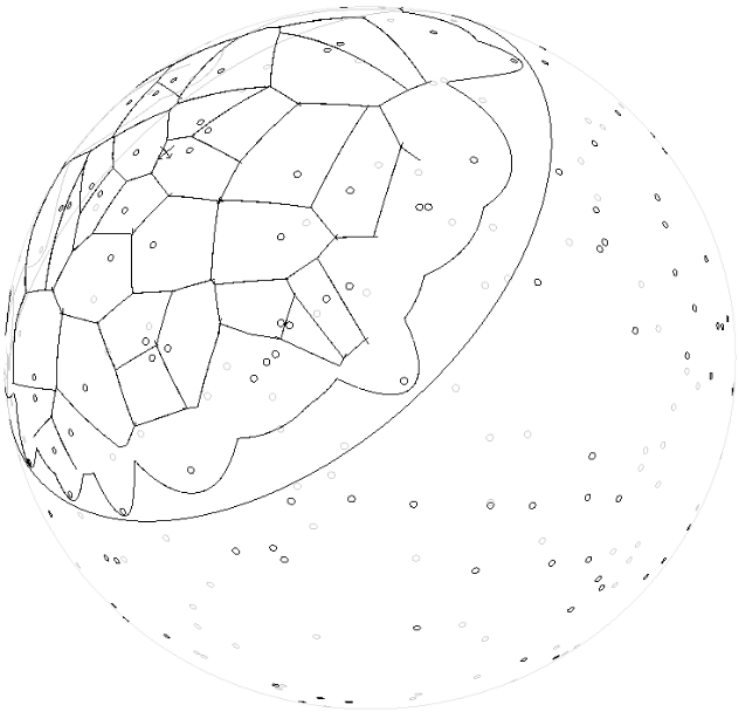


Figure 5.5: Fortune's algorithm process, sweeping-plane over a sphere.

Figure 5.5 shows a snap-shot of the Fortune's process. A fundamental difference with respect to the planar case is that the beach line on the sphere is a closed curve. As a consequence, the sweep must traverse the sphere twice—once on the exterior and once on the interior—to guarantee that all Voronoi vertices are detected. Despite this topological difference, the number of events remains linear in the number of sites, and the overall computational complexity of the algorithm is $\mathcal{O}(n \log n)$ in both time and memory, matching that of the planar Fortune's algorithm [89].

5.2.3 Coverage Control on the Hemisphere

We make use of the same performance function defined in 2.2.2, this has to be maximized in order to obtain the optimal coverage of the group of robots over $Q \subset S^2$.

Let $\phi : \mathbb{S}^2 \to \mathbb{R}_{\geq 0}$ denote a probability density function encoding the relative importance

of different viewing directions. The coverage objective is defined as:

$$\mathcal{H}(\mathcal{P}) = -\sum_{i=1}^{n} \int_{V_i(\mathcal{P})} \|q - p_i\|^2 \, \phi(q) \, dS, \tag{5.4}$$

where $V_i(\mathcal{P}) \subset S^2$ denotes the region of the hemispherical surface associated with agent i, and dS represents the surface area element.

The regions $V_i(\mathcal{P})$ are defined as Voronoi cells on the sphere with respect to the geodesic distance. Spherical Voronoi partitions form convex surface regions whose boundaries are arcs of great circles, providing a rigorous geometric decomposition of the manifold. This property allows the coverage objective (5.4) to be interpreted as a direct extension of classical planar coverage formulations to curved surfaces.

Intuitively, in the coverage problem, each agent is responsible for covering a subset of viewing directions, and the integral penalizes discrepancies between points on the hemisphere and the agent position representing that region. The density function ϕ enables biasing the coverage toward preferred directions—such as points of interest—while preserving a global notion of coverage quality over the admissible domain.

Configurations that locally maximize $\mathcal{H}(\mathcal{P})$ correspond to centroidal arrangements, in which each agent coincides with the density-weighted centroid of its associated spherical region. These centroidal configurations define the optimality condition for hemispherical coverage and constitute the target of the decentralized control strategies.

Maximizing the coverage functional (5.4) can be achieved through a variant of Lloyd's algorithm. As in the planar case, each iteration alternates between:

1. Updating the Voronoi partition induced by the current agent configuration $\mathcal{P}$, this is achieved with the Fortune's process described in 5.2.2,
2. Relocating each agent toward the centroid of its associated region.

5.2.3.1 Lloyd's Algorithm on the Hemisphere

Let $V_i(\mathcal{P})$ denote the Voronoi cell associated with agent i on the hemispherical surface. The density-weighted centroid is defined as

$$C_i = \frac{\int_{V_i} q \, \phi(q) \, dS}{\int_{V_i} \phi(q) \, dS}, \tag{5.5}$$

and the corresponding surface-consistent reference is obtained by radial projection onto the sphere,

$$\hat{C}_i = R \, \frac{C_i}{\|C_i\|}. \tag{5.6}$$

The Lloyd's step then moves the agent toward $\hat{C}_i$, which iteratively drives the configu-

ration toward a centroidal Voronoi tessellation on the hemisphere.

In the geometric (uniform) case $\phi(q) = 1$, a practical and commonly adopted approximation consists in computing the centroid using the polygonal representation of the cell. If V_i is represented by its ordered vertices $\{v_{i,1}, \ldots, v_{i,K_i}\} \subset \mathbb{R}^3$ (lying on the sphere of radius R), we approximate the centroid as:

$$\tilde{C}_i = \frac{1}{K_i} \sum_{k=1}^{K_i} v_{i,k}, \qquad \hat{C}_i = R \, \frac{\tilde{C}_i}{\|\tilde{C}_i\|}. \tag{5.7}$$

where $v_{i,k}$ is the k-th vetex of the polygon representing $\mathcal{V}_i$. This vertex-based centroid is efficient and numerically stable, and it captures the correct geometric behavior for uniform deployments on the hemispherical surface. It should be noted that in this representation the Voronoi diagram is defined as an approximation of the spherical domain, since the resulting diagram is composed of polyhedral elements, this remain still robust for the centroid computation task.

5.2.3.2 Lloyd's Algorithm under Gaussian Density Functions

To encode viewpoint preferences or points of interest, the density $\phi(q)$ can be selected to bias coverage toward desired directions. A typical choice is a Gaussian profile centered at a preferred viewing direction $q_0 \in S^2$, i.e.,

$$\phi(q) = \exp\left(-\frac{d_g(q, q_0)^2}{2\sigma^2}\right), \tag{5.8}$$

where $d_g(\cdot, \cdot)$ denotes the geodesic distance on the sphere scaled by the hemisphere radius—$d_g(\cdot) = R\, d(\cdot)$—, and σ standard deviation, can be used to control the spread of the preference.

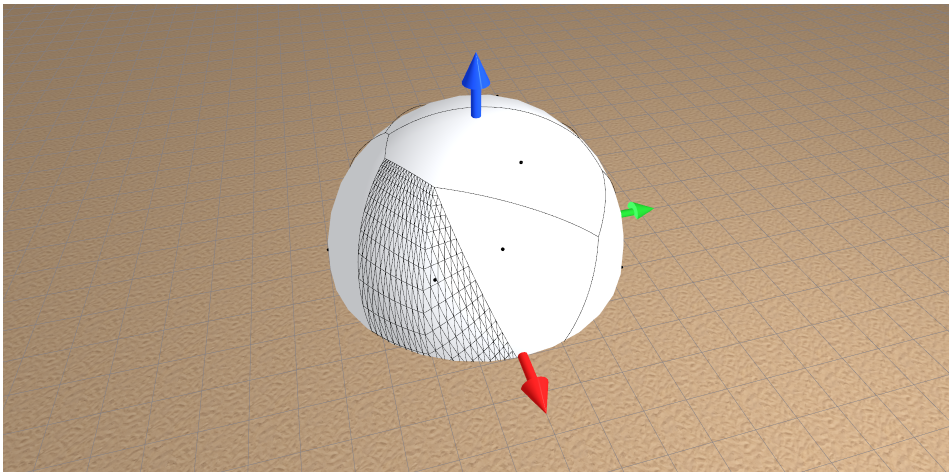


Figure 5.6: Gaussian-weighted centroid computation on a hemispherical Voronoi cell. The cell is fan–triangulated and each spherical triangle contribute to the centroid computation.

We introduce a numerical integration strategy consistent with the spherical geometry. Each Voronoi cell V_i is first represented as a spherical polygon and *fan–triangulated*

using one reference vertex, yielding spherical triangles that exactly tile the cell surface. Each spherical triangle is further subdivided into smaller spherical sub–triangles using a barycentric grid on the corresponding chordal simplex, followed by projection of the subdivision points back onto the sphere of radius R (Figure 5.6).

Consider a sub–triangle with unit vertices $u_1, u_2, u_3 \in \mathbb{S}^2$. Let a, b, and c denote the spherical side lengths—i.e. central angles—between the vertex pairs, and let $s = (a+b+c)/2$ be the semi-perimeter. The spherical excess E is obtained by L'Huilier's formula, and the exact area of the spherical sub–triangle is then:

$$A_{\triangle} = R^2 E. \tag{5.9}$$

Let $q_{\triangle}$ denote the projected centroid of the sub-triangle on the sphere. The contribution of this element to the cell centroid is weighted by the density and the exact spherical area, namely:

$$w_{\triangle} = \exp\left(-\frac{d_g(q_{\triangle}, q_{\mathrm{POI}})^2}{2\sigma^2}\right) A_{\triangle}. \tag{5.10}$$

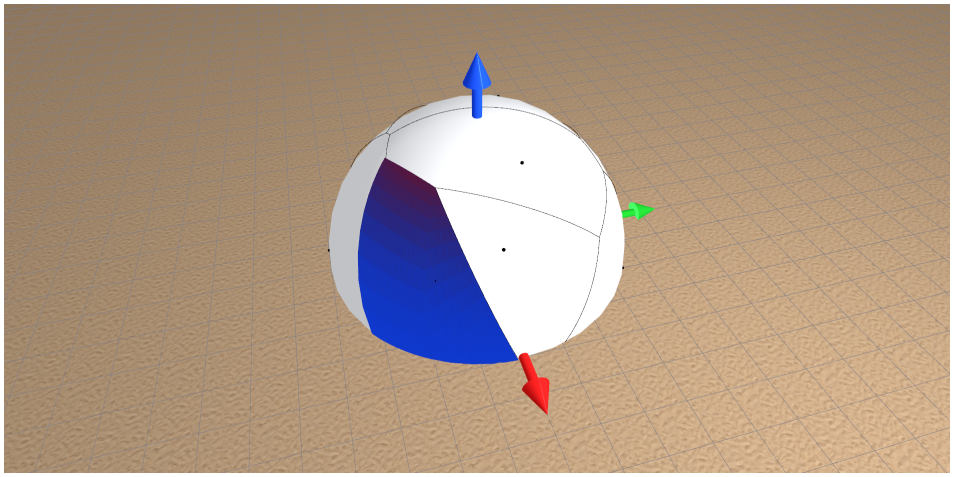


Figure 5.7: Gaussian-weighted centroid computation on a hemispherical Voronoi cell. Each sub–triangle contribute to the overall centroid computation via its weighted value on $\phi(q)$.

The cell centroid is finally approximated by accumulating the weighted contributions and normalizing:

$$\tilde{C}_i = \sum_{\triangle \in V_i} w_{\triangle}\, q_{\triangle}, \qquad \hat{C}_i = R\,\frac{\tilde{C}_i}{\|\tilde{C}_i\|}. \tag{5.11}$$

This procedure preserves the intrinsic geometry of the sphere (through exact spherical areas) while allowing flexible, density-aware centroid estimation. Exact area expressions for spherical cases relate to classical results on spherical geometry and higher-dimensional analogs; see, e.g., [90].

5.2.3.3 Computational Complexity Analysis

Algorithm 2 Lloyd's Algorithm on the Sphere

Require: $\mathcal{P} = \{p_1, \ldots, p_n\} \subset S^2$, density $\phi : S^2 \to \mathbb{R}_+$

for $k = 1, 2, \ldots$ **do**
  Voronoi Diagram $\mathcal{V}(\mathcal{P}) \leftarrow$ spherical sweep [89]
  Sort $\mathcal{P}$ by latitude, process site/circle events
  Maintain beach curve, update cells $\mathcal{V}_i \subset S^2$
  for $i = 1$ to n **do**
    $C_i \leftarrow \dfrac{\int_{\mathcal{V}_i} q\,\phi(q)\,dS}{\int_{\mathcal{V}_i} \phi(q)\,dS}$ // *weighted centroid*
    $p_i \leftarrow \dfrac{C_i}{\|C_i\|}$ // *normalize to* S^2
  end for
end for
return $\mathcal{P}$

As shown in Algorithm 2 each Lloyd's iteration on the hemisphere consists of two dominant steps: (i) Voronoi construction on the sphere and (ii) centroid computation for each cell.

The Voronoi update is performed using the spherical sweep-plane method of Zheng *et al.* [89], which preserves the asymptotic complexity of the planar Fortune's algorithm. The construction requires $\mathcal{O}(n \log n)$ time and $\mathcal{O}(n)$ memory for n sites, since the number of site and circle events is linear and each event can be processed using balanced search structures and a priority queue.

The centroid phase depends on the numerical resolution used for integration. In the uniform case, the vertex-averaging approximation (5.7) incurs a cost proportional to the number of vertices per cell; in practice, the average number of polygon vertices is bounded, so the total centroid cost is near-linear in n. In the Gaussian case, if each cell is decomposed into a bounded number of sub–triangles (or more generally if a fixed integration budget m is used per cell), the centroid computation scales as $\mathcal{O}(nm)$.

Therefore, the total complexity per Lloyd's iteration is:

$$\mathcal{O}(n \log n) + \mathcal{O}(nm), \tag{5.12}$$

which reduces to $\mathcal{O}(n \log n)$ when the per-cell integration budget is bounded (i.e., $m = \mathcal{O}(1)$). This preserves the favorable scalability of planar Voronoi-based coverage while enabling coverage control directly on a hemispherical manifold.

5.3 Applied Hemispherical Coverage Control

In this Section we present the practical implementation and the results of the hemispherical contour introduced in Section 5.2.

We first describe the system setup and experimental platforms used in simulation and field trials. We then detail the experimental protocol, including evaluation metrics and test procedures. Finally, we report and discuss the results obtained in simulation and real-world experiments, highlighting the correspondence between theoretical predictions and observed system behavior.

Figure 5.8: Eight drones in a hemispherical configuration during an experiment.

5.3.1 Test and Setups

The aim of the experiments presented in this section is to evaluate the performance of a team of unmanned aerial vehicles operating under the decentralized hemispherical coverage control strategy introduced in Section 5.2. The objective is to assess the effectiveness, robustness, and scalability of the proposed approach when deployed in realistic conditions, where sensing, communication, and actuation are subject to non-idealities.

To this end, both high-fidelity dynamic simulations and real-world field experiments were conducted. Simulation trials were used to explore a wide range of configurations and team sizes under controlled conditions, enabling statistically meaningful performance evaluation. Complementary field experiments were carried out in a large outdoor environment to validate the practical feasibility of the approach and to assess its behavior when integrated with real UAV platforms and onboard flight control systems.

The following subsections describe the experimental platforms, control architecture, and evaluation protocols adopted for simulation and field testing, providing the necessary context for the results discussed later in this section. An extended experimental evaluation of decentralized hemispherical coverage with aerial robots is reported in [6].

5.3.2 Experimental Protocol

The hemispherical coverage control strategy was implemented using the C++ variant of the ROS [43] framework and deployed consistently across both simulation and real-world experiments. This unified software architecture ensured that the same high-level control logic, communication interfaces, and execution flow were preserved across experimental settings, enabling a meaningful comparison between simulated and physical results. In addition to the coverage control framework, selected experiments incorporated onboard visual perception through a forward-facing camera. The camera feed was processed in real time using object recognition techniques to detect and localize objects of interest in the environment. The resulting detections were used to bias the camera pointing direction toward salient regions, enabling perception-driven attention while preserving the underlying coverage behavior. Object detection was implemented using modern YOLO-based architectures, which provide a favorable trade-off between detection accuracy and computational efficiency for real-time deployment on aerial platforms [91].

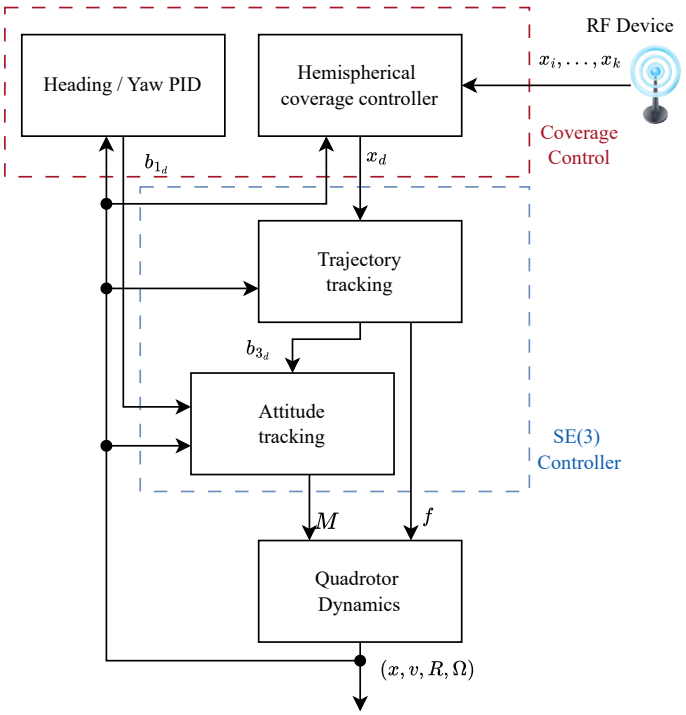


Figure 5.9: The control architecture used for the decentralized hemispherical coverage. The position reference x_d on the manifold is computed using RF-received neighbor states $\{x_i, \ldots, x_k\}$. A dedicated *Heading/Yaw PID* generates the desired body-axis direction $\mathbf{b}_{1d}$, ensuring viewpoint toward the center or the target is given. These two high-level guidance signals feed into the controller-a geometric SE(3), which performs trajectory and attitude tracking to produce thrust f and moments M. The quadrotor dynamics return the estimated state (x, v, R, Ω) to all upstream modules.

5.3.2.1 Simulation Protocol

Simulation experiments were conducted using a high-fidelity SITL setup based on the Gazebo simulator integrated with the PX4 autopilot stack [92]. This configuration allowed realistic modeling of vehicle dynamics, sensor feedback, and control loops while maintaining full access to internal states for logging and analysis. Each UAV was executed within a dedicated simulation instance, with the coverage controller, LLC, inter-agent communication, and PX4 SITL running concurrently.

The execution of each simulation trial followed a structured multi-phase protocol designed to ensure safety, repeatability, and consistency across runs. Specifically, each experiment was composed of the following phases:

1. Autonomous take-off to a predefined altitude;
2. Planar coverage motion used to spatially separate the agents and establish a collision-free initial configuration;
3. Activation of the hemispherical coverage controller, during which agents converged to the desired spherical configuration around the target;
4. Reversion to planar coverage to safely disengage from the hemispherical constraint;
5. Autonomous landing.

Although collision avoidance is inherently supported by the Voronoi-based spatial partitioning of the coverage algorithm, this phased execution strategy was deliberately adopted to maintain controlled transitions between flight regimes and to mitigate potential risks during initialization and termination of each run.

Figure 5.10: Left: quadrotor model used in the Gazebo-based simulation environment. Right: example of simulated onboard camera views during a multi-drone hemispherical coverage task.

5.3.2.2 Field Test Protocol

Field experiments were conducted at a remote outdoor test site in the Al Dhafra region of the Abu Dhabi Emirate, United Arab Emirates. The UAV platform employed was a custom-designed quadrotor developed at the TII, featuring a 60 cm carbon-fiber airframe, a Pixhawk 6X flight controller, and an NVIDIA Jetson Orin NX onboard computer

responsible for executing the mission logic and coverage control algorithms. Inter-UAV communication was achieved through a 2.4 GHz radio-frequency link.

The same control software stack used in simulation was deployed onboard the real vehicles, ensuring consistency between experimental modalities. To guarantee persistent visual focus on the target, the yaw orientation of each UAV was actively regulated by augmenting the control state with the yaw angle ψ. A proportional–integral–derivative (PID) controller governed the first-order yaw dynamics, enabling the joint regulation of translational velocity $\dot{\mathbf{p}}$ and yaw rate $\dot{\psi}$.

In contrast to the simulated environment, real-world experiments imposed additional safety constraints. In particular, the translational speed of each vehicle was capped at a maximum ground velocity of 8 m/s. As in simulation, each field trial followed a structured execution pipeline consisting of take-off, planar coverage, hemispherical coverage, planar disengagement, and landing. This procedure ensured safe transitions between flight phases and reduced the likelihood of trajectory intersections during transient maneuvers.

While operating on the hemispherical surface, collision avoidance was inherently enforced by the Voronoi-based spatial partitioning of the coverage controller. The additional procedural constraints were introduced solely to improve operational safety and repeatability during real-world deployments.

Figure 5.11: Left: one of the quadrotor platforms used in the field experiments. Right: snapshot of an eight-drone hemispherical coverage experiment during execution.

5.3.2.3 Metrics

The performance of the proposed hemispherical coverage strategy is evaluated using two complementary metrics. The first metric is derived directly from the coverage objective function introduced in Section 5.2.3, while the second quantifies convergence toward a centroidal Voronoi configuration. Together, these metrics capture both the global coverage quality and the geometric optimality of the agent configuration.

Coverage Objective Value. The primary performance indicator is the value of the coverage objective function $\mathcal{H}(\mathcal{P})$ defined in (5.4). This functional measures the density-weighted quality of coverage over the hemispherical domain, penalizing the distance between points on the surface and their assigned agents. Higher values of $\mathcal{H}(\mathcal{P})$ correspond to improved coverage performance, accounting for both the spatial distribution of

agents and the underlying importance density $\phi(q)$. Monitoring the temporal evolution of $\mathcal{H}(\mathcal{P})$ allows assessing convergence behavior and comparing steady-state performance across different team sizes and density profiles.

CVT Error. In addition to the global objective, we evaluate the geometric notion of *configuration optimality* [2, 24], which quantifies how close the current configuration is to a CVT. In an ideal CVT, each site coincides with the centroid of its associated Voronoi cell, i.e., $\mathbf{p}_i = \mathbf{c}_i$ for all agents i, this remains true also for the hemispherical domain.

To measure deviations from this condition, we define the per-agent centroid displacement:

$$E_i = \|\mathbf{p}_i - \mathbf{c}_i\|, \tag{5.13}$$

and aggregate it into a configuration-level metric:

$$E_{\mathrm{CVT}}(\mathcal{P}) = \sum_{i=1}^{n} E_i, \tag{5.14}$$

which has values in the $\mathbb{R}^+$ domain and vanishes if and only if the configuration is centroidal. This metric provides a direct geometric measure of convergence toward the optimal Lloyd's equilibrium, independently of the specific density function $\phi(q)$.

While $\mathcal{H}(\mathcal{P})$ captures the task-oriented coverage quality, $E_{\mathrm{CVT}}(\mathcal{P})$ reflects the internal geometric consistency of the Voronoi-based deployment. Evaluating both metrics jointly enables a comprehensive assessment of the proposed control strategy, highlighting both its effectiveness in optimizing the coverage objective and its ability to achieve centroidal configurations on the hemispherical manifold.

5.3.3 Results

5.3.3.1 Simulation Results

Simulation experiments were conducted to evaluate convergence behavior, scalability, and the influence of density functions on the resulting hemispherical configurations. Figure 5.12 reports the temporal evolution of the coverage objective $\mathcal{H}(\mathcal{P})$, averaged over ten independent runs. After an initial transient corresponding to take-off and dispersion, the objective value increases monotonically and stabilizes once the agents reach a steady hemispherical configuration. Unlike the results presented in Chapter 3, the values of the coverage objective function are normalized in this case. At this stage of the analysis, the effects of sensing range and covered area size are not explicitly considered; instead, the focus is placed directly on the system dynamics. It can be observed that configurations with a smaller number of drones exhibit a faster transient response of the coverage function $\mathcal{H}(\mathcal{P})$. This behavior is primarily due to the velocity saturation introduced for safety reasons, which limits how quickly the agents can converge toward the centroidal Voronoi tessellation optimum.

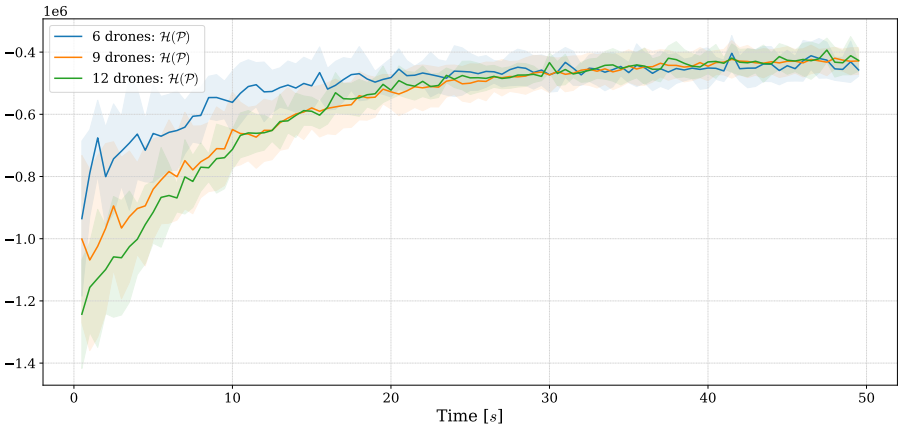


Figure 5.12: Temporal evolution of the coverage objective $\mathcal{H}(\mathcal{P})$, averaged over ten simulation runs. Solid lines indicate the mean, while shaded regions represent one standard deviation.

Convergence toward centroidal configurations is further quantified by the CVT error metric $E_{\mathrm{CVT}}(\mathcal{P})$. Figure 5.13 shows that the error decreases consistently over time and converges toward small steady-state values for teams of 6, 9, and 12 drones on a hemisphere of radius 15 m. These results confirm that the Lloyd-based update drives the system toward centroidal Voronoi equilibria on the spherical surface.

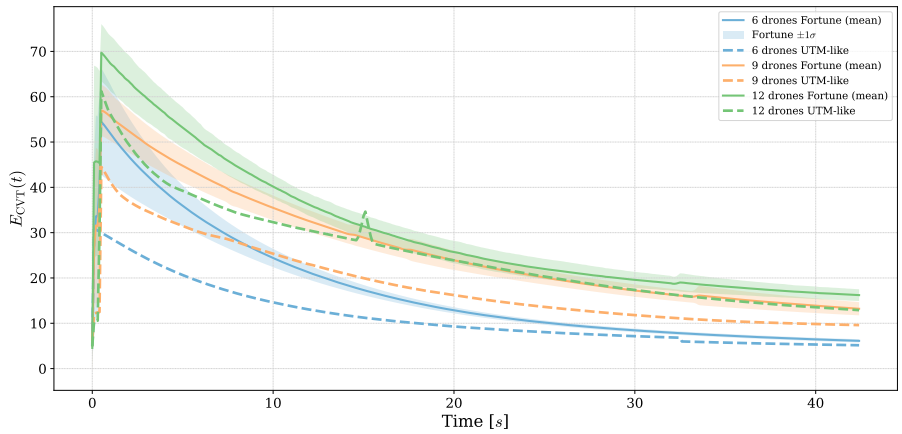


Figure 5.13: CVT error $E_{\mathrm{CVT}}(\mathcal{P})$ for teams of 6, 9, and 12 drones in simulation on a hemisphere of radius 15 m.

Representative agent trajectories are illustrated in Figure 5.14. Subfigures (a)–(c) show geometric (uniform-density) hemispherical coverage, where agents converge to approximately uniform angular spacing as the team size increases. Subfigures (d)–(f) depict Gaussian-biased coverage, where the final configurations concentrate around regions of higher density. Varying the Gaussian width σ highlights the ability of the proposed framework to smoothly adapt the formation in response to user-defined importance profiles, without compromising stability or convergence.

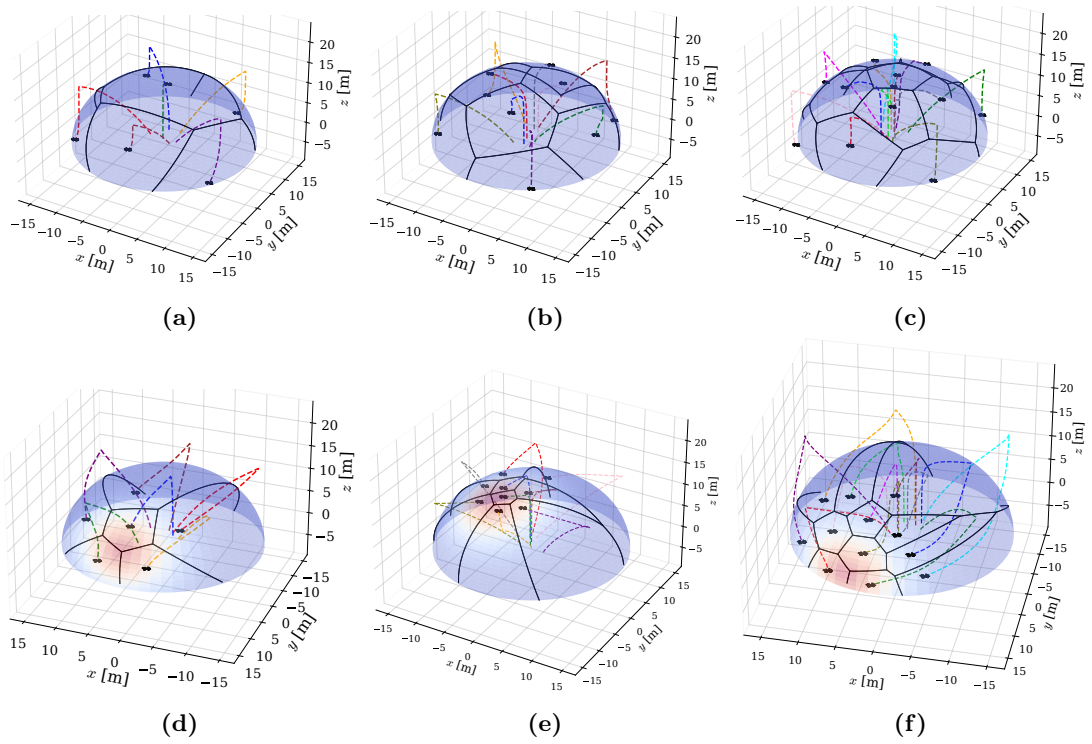


Figure 5.14: Hemispherical coverage results in simulation. In (a), (b), and (c) geometric hemisphere coverage with teams of 6, 9, and 12 drones, respectively. Sub-figures (d), (e), and (f) illustrate Gaussian-biased hemisphere coverage for teams of 6, 9, and 12 drones with different Gaussian centers on the spherical surface. Varying values of σ highlight how the drone configurations adapt to different density distributions.

5.3.3.2 Field Experiments Results

Field experiments were conducted to validate the proposed hemispherical coverage strategy under real-world conditions, including sensing noise, communication delays, wind disturbances, and actuator limitations. A fleet of eight quadrotors was deployed in an outdoor environment following the experimental protocol described in Section 5.3.2.2.

Figure 5.15 shows the reconstructed trajectories for two representative experiments. In the geometric case, the drones converge to a uniformly distributed hemispherical configuration around the target, closely mirroring the behavior observed in simulation. In the Gaussian-biased case, the formation adapts toward the region of higher density, demonstrating that the density-aware centroid computation remains effective when deployed on physical platforms.

The temporal evolution of a complete experimental run is illustrated in Figure 5.16. The sequence highlights the structured execution pipeline: take-off, initiation of hemispherical coverage, steady-state operation, and adaptation to a reduced hemisphere radius. Despite the dynamic changes in constraints, the formation remains coherent and collision-free, confirming the robustness of the decentralized control strategy.

Overall, the field results confirm that the proposed hemispherical coverage framework transfers reliably from simulation to real-world deployment, achieving stable decentralized coordination under both uniform and density-biased configurations.

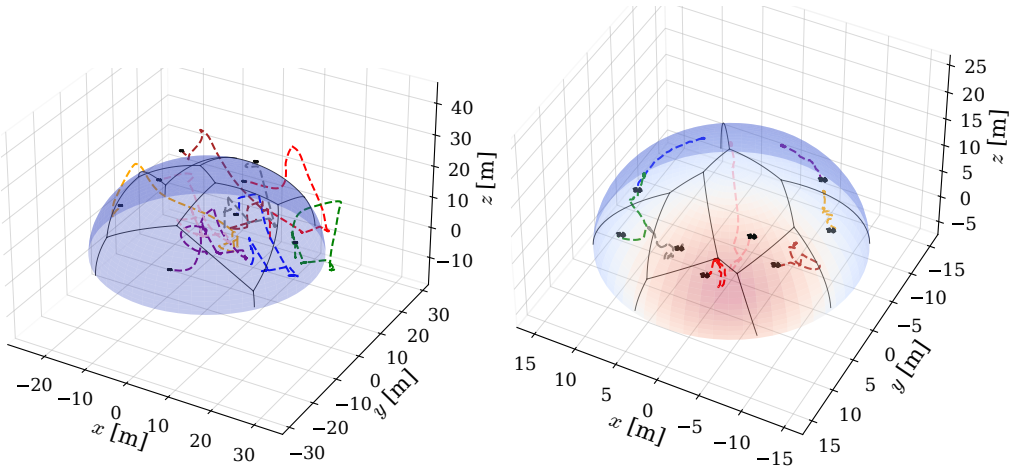


Figure 5.15: Real-world hemispherical coverage experiments with eight drones. Left: geometric (uniform-density) coverage. Right: Gaussian-biased coverage.

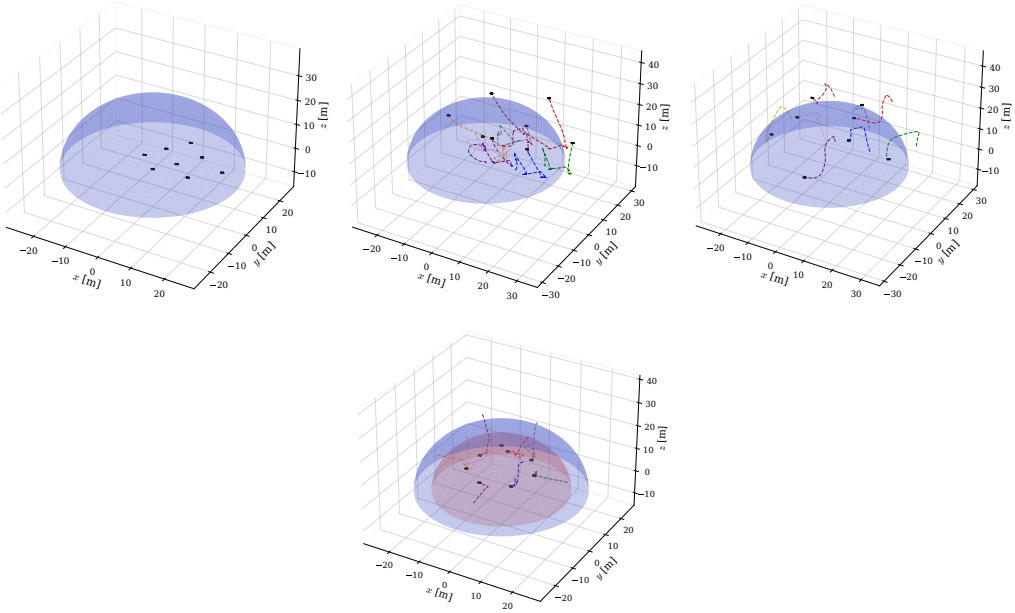


Figure 5.16: Evolution of a representative field experiment with eight drones. From left to right: take-off and initial dispersion; activation of hemispherical coverage; steady-state operation; adaptation to a reduced hemisphere radius.

5.3.3.3 Methodology Comparison

As a baseline of comparable computational complexity, we also evaluate performance using a second coverage method: let $(x, y, z) \in \mathbb{R}^3$ lie on a sphere of radius R with geographic coordinates (latitude ϕ, longitude λ). We project to $(\bar{x}, \bar{y}) \in \mathbb{R}^2$ via an equirectangular-like mapping referenced at (ϕ_0, λ_0):

$$\bar{y} = (\phi - \phi_0)\, R, \qquad \bar{x} = (\lambda - \lambda_0)\, R\, \cos(\phi_0). \tag{5.15}$$

This parameterization allows us to treat $(\bar{x}, \bar{y})$ as locally planar for planning and coverage control. Fixed $\phi = \phi_0 + \frac{\bar{y}}{R}$, $\lambda = \lambda_0 + \frac{\bar{x}}{R\cos(\phi_0)}$ the inverse map is:

$$x = R\cos\phi\,\cos\lambda, \qquad y = R\cos\phi\,\sin\lambda, \qquad z = R\sin\phi. \tag{5.16}$$

Geographic coordinate transformations are performed using the Universal Transverse Mercator projection, originally formalized in [93]. As with any equirectangular construction, this mapping is neither conformal nor equal-area; metric distortion grows with $|\phi - \phi_0|$ and away from the reference meridian λ_0. In practice, choosing (ϕ_0, λ_0) near the operational footprint keeps distortions small, making the planar proxy suitable for local coverage on the hemisphere. From an intuitive standpoint, this offers a simple and immediate baseline to compare the proposed algorithm against a methodology with the same asymptotic complexity: the $(\bar{x}, \bar{y}) \in \mathbb{R}^2$ domain is used to compute planar Voronoi diagrams using Fortune's sweepline algorithm in $O(n \log n)$.

Figure 5.13 further quantifies this comparison by reporting the CVT error $E_{\mathrm{CVT}}(\mathcal{P})$ for configurations with 6, 9, and 12 drones under both methodologies, this second metodology is reported as *UTM-like* due to the similarity with Universal Transverse Mercator projected reference frame. When using the coordinate-transformation approach (5.15), the centroids computed in the planar domain can be mapped back exactly onto the hemispherical surface, resulting in negligible residual error once expressed in $\mathbb{R}^3$.

In contrast, the sweep-plane approach constructs the Voronoi diagram through a polyhedral approximation of the spherical surface, which introduces a small off-surface component in the centroid computation. This effect leads to a higher measured CVT error when evaluated in Euclidean space. Nevertheless, the error decreases asymptotically as the number of agents increases, indicating convergence toward centroidal configurations and confirming the validity of the approximation.

Importantly, while the coordinate-transformation method achieves lower numerical error in this metric, it remains sensitive to *geographic distortions* and relies on the selection of a suitable reference frame. The sweep-plane formulation, by operating intrinsically on the spherical manifold, avoids such distortions and provides consistent geometric behavior over the entire hemisphere. As a result, the proposed approach offers a more principled and globally consistent solution for hemispherical coverage, while maintaining competitive convergence properties and scalability.

5.4 Discussion

This Chapter investigated decentralized coverage control on hemispherical manifolds, bridging the gap between higher-dimensional coverage theory and practical multi-UAV deployment. By formulating coverage directly on the spherical surface and leveraging Voronoi-based partitioning with a Fortune-inspired sweep-plane construction, the proposed methodology preserves the computational efficiency of planar coverage algorithms while addressing the intrinsic geometric constraints of viewpoint-oriented tasks.

The simulation results demonstrate that the proposed control strategy consistently con-

verges toward centroidal configurations under both uniform and Gaussian density functions. The evolution of the coverage objective $\mathcal{H}(\mathcal{P})$ and the decay of the CVT error $E_{\mathrm{CVT}}(\mathcal{P})$ confirm that the Lloyd-based descent remains effective on the hemispherical manifold. Increasing the number of agents improves coverage quality and reduces geometric error, highlighting the scalability of the approach and its suitability for larger teams.

Field experiments further validate these findings under real-world conditions. Despite the presence of communication delays, wind disturbances, sensing noise, and actuator constraints, the UAV team successfully achieved decentralized hemispherical deployment in all tested scenarios. The consistency between simulation and experimental results indicates that the proposed formulation is not only theoretically sound but also robust to non-idealities inherent to outdoor aerial robotics. The staged execution strategy—comprising take-off, planar dispersion, hemispherical coverage, and landing—proved effective in ensuring safety and repeatability without compromising the intrinsic collision-avoidance properties of the Voronoi-based controller.

The methodology comparison highlights an important trade-off between geometric fidelity and numerical precision. While planar coordinate-transformation approaches yield lower CVT error when evaluated in Euclidean space, they inherently suffer from metric distortions and dependence on reference-frame selection. In contrast, the sweep-plane spherical Voronoi construction operates directly on the manifold, providing globally consistent behavior at the cost of a mild polyhedral approximation. The observed asymptotic convergence of the CVT error confirms that this approximation does not hinder practical performance, while offering superior robustness and interpretability for hemispherical coverage tasks.

Overall, the results support the central claim of this chapter: hemispherical coverage can be achieved in a fully decentralized manner with $\mathcal{O}(n \log n)$ complexity, minimal communication, and compatibility with standard robotic software stacks. The proposed framework provides a principled and deployable solution for viewpoint-driven multi-UAV applications such as inspection, cinematography, and cooperative perception.

Chapter 6

Conclusive Remarks

6.1 Main Findings

This thesis investigated decentralized coverage control for multi-UAV systems with the objective of assessing how Voronoi-based methodologies can be reliably deployed in realistic operational scenarios. Through the combined development of control strategies, algorithmic extensions, simulation and real-world experimental validation, several key findings emerge.

A first central finding is that distributed coverage control based on Voronoi partitioning remains effective under severe sensing and communication constraints, provided that the control architecture is designed to explicitly account for locality. By relying exclusively on limited-range information exchange and locally computable Voronoi regions, the proposed strategies eliminate the need for global environment knowledge or centralized coordination. Experimental results demonstrate that such decentralized formulations can scale to multi-UAV teams while maintaining stable and coherent group behavior, even in outdoor environments affected by sensing noise, actuation limits, and asynchronous updates. This confirms that fully distributed coverage control is not only theoretically sound but also practically deployable on real aerial platforms.

A second major finding concerns the role of time variation in coverage objectives. When spatial importance evolves over time, classical Lloyd-based formulations must be reinterpreted not as convergence problems but as tracking problems, in which agents continuously follow moving centroidal references. Both simulations and field experiments show that proportional Lloyd's controllers, while effective in static settings, can lead to oscillations, overshoot, and loss of coherence when applied to real UAV systems under dynamic conditions. Introducing smooth, state-dependent control structures significantly improves stability and responsiveness, enabling the team to track time-varying density distributions with bounded degradation in coverage quality. These results highlight that, in time-varying scenarios, control law design plays a more critical role than partitioning accuracy itself. In the scenarios considered in this work, time variation in the coverage objective is not imposed exogenously, but rather emerges from the interaction between onboard sensing, cooperative perception, and the distributed reconstruction of the envi-

ronment. In particular, spatial information is generated through a perception pipeline in which each UAV acquires local measurements using onboard sensors and shares partial observations with neighboring agents. These measurements are fused through cooperative triangulation procedures to estimate the position of targets or events of interest, and the resulting estimates are mapped onto the coverage domain through time-varying probability density functions. Gaussian density models are employed to encode both the estimated target location and the associated uncertainty, yielding a coverage objective that evolves continuously as new sensory information becomes available. This formulation ensures coherence between the sensing process, the reconstructed representation of the environment, and the definition of the coverage functional optimized by the control law.

The adoption of Gaussian-weighted densities further enables a principled coupling between perception quality and coverage behavior. Variations in sensing confidence, arising for instance from limited field of view, geometric degeneracies in triangulation, or intermittent detections, are directly reflected in the shape and spread of the density function. As a result, the centroidal references tracked by the UAVs adapt not only to target motion but also to the reliability of the reconstructed world model. Experimental results confirm that this perception-driven, density-based formulation allows the multi-UAV system to reallocate sensing effort dynamically, concentrating resources in regions of higher inferred importance while maintaining global coverage consistency. This tight integration between sensing, distributed world reconstruction, and coverage control reinforces the interpretation of time-varying coverage as a closed-loop, information-driven tracking problem, rather than a purely geometric partitioning task.

A further finding of this work is that safety and coordination constraints can be systematically embedded into coverage control architectures without compromising decentralization. By integrating Control Barrier Functions and Control Lyapunov Functions at the control layer, collision avoidance, obstacle avoidance, and formation maintenance objectives can be enforced alongside coverage optimization. The resulting dual-layer architectures preserve the distributed nature of the coverage strategy while ensuring safe execution on physical platforms. Experimental validation confirms that prioritizing safety constraints does not prevent convergence toward meaningful coverage configurations, but rather enables reliable operation in cluttered and uncertain environments.

One of the key contributions of this thesis is the extension of coverage control methodologies to non-planar domains, with a specific focus on hemispherical environments. Coverage on curved surfaces introduces non-trivial geometric and algorithmic challenges, including the definition of appropriate distance metrics, centroid computation on manifolds, and efficient partitioning. By developing a spherical variant of Fortune's algorithm and density-aware centroid computation on hemispherical surfaces, this work demonstrates that Voronoi-based coverage can be generalized beyond planar domains with tractable computational complexity. Simulation and real-world experiments with multiple UAVs validate the feasibility and scalability of the proposed approach, supporting both uniform and Gaussian-weighted coverage configurations on curved manifolds.

Finally, an overarching finding of this thesis is the central role played by experimental

validation in shaping algorithmic and control design choices. Real-world experiments consistently exposed limitations that are not apparent in idealized simulations, such as sensitivity to noise, saturation effects, and transient instabilities. These observations directly motivated modifications to nominal control laws and the adoption of smoother, constraint-aware formulations. As a result, experimental feedback was not treated as a post-hoc validation step, but as an integral component of the research process. This experimental-driven approach strengthens the relevance of the proposed methodologies and reinforces their applicability to real multi-UAV deployments.

Overall, the results presented in this thesis demonstrate that decentralized coverage control can be transformed from a primarily theoretical construct into a robust, scalable, and experimentally validated coordination strategy for multi-UAV systems operating in dynamic and geometrically complex environments.

6.2 Future Works

The results presented in this thesis open several directions for future research aimed at further extending decentralized coverage control toward increasingly realistic and complex operational scenarios. While the proposed methodologies demonstrate robustness, scalability, and practical deployability, they also highlight a number of open challenges that naturally emerge when moving from idealized formulations toward fully autonomous multi-UAV systems operating in dynamic and uncertain environments.

A first direction concerns the refinement of coverage control laws beyond classical first-order Lloyd's descent. Although this work has shown that smoothing and state-dependent gain modulation significantly improve stability in time-varying scenarios, first-order dynamics remain sensitive to delays, actuator saturation, and communication constraints. Future research may investigate alternative formulations explicitly designed to account for these non-idealities, as well as second-order or momentum-based coverage dynamics that incorporate velocity states or inertial effects. The coverage strategies developed in this thesis are based on Lloyd-type first-order descent dynamics, which can be written in the general form $\dot{p}_i = k_i(c_i - p_i)$, although this formulation is simple and naturally decentralized, it does not explicitly account for inertial effects, actuator limitations, or delayed responses. A natural extension for future research is the study of second-order coverage dynamics, in which the agent velocity is explicitly modeled, for example as $\ddot{p}_i + d_i\dot{p}_i = k_i(c_i - p_i)$, with $d_i > 0$ a damping coefficient. Such formulations may better capture the dynamics of real aerial platforms and improve transient behavior. Future work should investigate whether these more general control laws can preserve the decentralization, stability, and convergence properties of classical Lloyd-based coverage, especially in the presence of communication delays, actuator saturation, and time-varying density functions. Such approaches have the potential to improve transient performance, reduce oscillatory behavior, and increase energy efficiency, while preserving the decentralized nature of the coverage strategy

A second promising direction involves the extension of coverage methodologies to more

general geometric domains. The hemispherical coverage framework developed in this thesis represents a first step toward manifold-aware coordination on curved surfaces. Future work may generalize these concepts to arbitrary curved surfaces represented by meshes or implicit functions, as well as to fully three-dimensional volumetric domains. This would enable coverage of complex structures such as building facades, industrial plants, or natural environments, and would require the development of distributed partitioning and centroid computation techniques compatible with general surface representations and non-Euclidean metrics.

Another important avenue of research concerns the explicit integration of sensing constraints into the coverage formulation. While this thesis already couples coverage objectives with perception-driven density functions, future extensions may incorporate limited field of view, occlusions, visibility constraints, and obstacle-induced information loss directly into the coverage objective. Moving from open environments to cluttered urban or indoor scenarios would allow the system to reason explicitly about sensing quality and information gain, further strengthening the link between coverage control and active perception.

Finally, learning-based methods represent a complementary direction for extending the proposed framework. Rather than replacing analytical control laws with end-to-end learned policies, learning could be used to augment specific components of the perception–control pipeline that are difficult to model explicitly. Examples include learning spatial density functions from raw sensor data, adapting control gains based on observed performance, or predicting the evolution of points of interest under partial observability. Integrating such learning mechanisms within a structure-preserving framework would allow the system to retain safety, stability, and decentralization guarantees, while improving adaptability and performance in complex and evolving environments.

Bibliography

[1] J. Cortés, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 243–255, 2004.

[2] F. Bertoncelli, M. Belal, D. Albani, F. Pratissoli, and L. Sabattini, "On limited-range coverage control for large-scale teams of aerial drones: Deployment and study," in *The 16th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2022.

[3] M. Belal, D. Albani, and L. Sabattini, "Understanding the role of time-varying targets in adaptive distributed area coverage control," in *International Symposium on Experimental Robotics*. Springer, 2023, pp. 239–249.

[4] M. Catellani, M. Belal, and L. Sabattini, "Dual-layer control architecture for cooperative environmental monitoring in multi-robot systems," in *The 17th International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2024.

[5] M. Belal, T. Manoni, D. Albani, and L. Sabattini, "Decentralized multi-robot coverage of hemispherical surfaces via fortune-based partitioning," in *Submitted to the 15th International Conference on Swarm Intelligence (ANTS)*, 2026.

[6] ——, "Experimental evaluation of decentralized hemispherical coverage for multi-uav systems," in *Proceedings of The International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2026.

[7] K. Nagatani, S. Kiribayashi, Y. Okada, K. Otake, K. Yoshida, S. Tadokoro, T. Nishimura, T. Yoshida, E. Koyanagi, M. Fukushima, *et al.*, "Emergency response to the nuclear accident at the fukushima daiichi nuclear power plants using mobile rescue robots," *Journal of Field Robotics*, vol. 30, no. 1, pp. 44–63, 2013.

[8] F. Greenwood, E. L. Nelson, and P. G. Greenough, "Flying into the hurricane: A case study of uav use in damage assessment during the 2017 hurricanes in texas and florida," *PLoS one*, vol. 15, no. 2, p. e0227808, 2020.

[9] I. A. Hameed, "A coverage planner for multi-robot systems in agriculture," in *2018 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE, 2018, pp. 698–704.

[10] A. Ribeiro and J. Conesa-Muñoz, "Multi-robot systems for precision agriculture," in *Innovation in Agricultural Robotics for Precision Agriculture*. Springer, 2021, pp. 151–175.

[11] J. Kim, S. Kim, C. Ju, and H. I. Son, "Unmanned aerial vehicles in agriculture: A review of perspective of platform, control, and applications," *IEEE Access*, vol. 7, pp. 105 100–105 115, 2019.

[12] Y. Fan, Y. Hu, L. Jiang, Q. Liu, L. Xiong, J. Pan, W. Hu, Y. Cui, T. Chen, and Q. Zhang, "Intelligent disinfection robots assist medical institutions in controlling environmental surface disinfection," *Intelligent Medicine*, vol. 1, no. 01, pp. 19–23, 2021.

[13] Federal Aviation Administration, "Summary of small unmanned aircraft rule (part 107)," Online; accessed 2024-12-01, 2023, https://www.faa.gov/uas/commercial_operators/part_107_rule.

[14] European Union Aviation Safety Agency, "U-space and Urban Air Mobility," Online; accessed 2024-12-01, 2021, https://www.easa.europa.eu/en/domains/u-space.

[15] V. Kumar and N. Michael, "Opportunities and challenges with autonomous micro aerial vehicles," *The International Journal of Robotics Research*, vol. 31, no. 11, pp. 1279–1291, 2012.

[16] C. W. de Silva, "Some issues and applications of multi-robot cooperation," in *2016 IEEE 20th International Conference on Computer Supported Cooperative Work in Design (CSCWD)*. IEEE, 2016, pp. 2–2.

[17] J. Cortés and M. Egerstedt, "Coordinated control of multi-robot systems: A survey," *SICE Journal of Control, Measurement, and System Integration*, vol. 10, no. 6, pp. 495–503, 2017.

[18] W. Ren and N. Sorensen, "Distributed coordination architecture for multi-robot formation control," *Robotics and Autonomous Systems*, vol. 56, no. 4, pp. 324–333, 2008.

[19] S. Lloyd, "Least squares quantization in pcm," *IEEE transactions on information theory*, vol. 28, no. 2, pp. 129–137, 1982.

[20] A. Okabe, "Spatial tessellations," *International Encyclopedia of Geography: People, the Earth, Environment and Technology: People, the Earth, Environment and Technology*, pp. 1–11, 2016.

[21] S. Fortune, "A sweepline algorithm for voronoi diagrams," in *Proceedings of the second annual symposium on Computational geometry*, 1986, pp. 313–322.

[22] R. Soukieh, I. Shames, and B. Fidan, "Obstacle avoidance of non-holonomic unicycle robots based on fluid mechanical modeling," in *Proceedings of the European Control Conference*, 2009.

[23] D. Lee, A. Franchi, H. Son, C. Ha, H. Bulthoff, and P. Robuffo Giordano, "Semi-autonomous haptic teleoperation control architecture of multiple unmanned aerial vehicles," *IEEE/ASME Transactions on Mechatronics*, vol. 18, no. 4, pp. 1334–1345, Aug 2013.

[24] J. Cortés, S. Martinez, and F. Bullo, "Spatially-distributed coverage optimization and control with limited-range interactions," *ESAIM: Control, Optimisation and Calculus of Variations*, vol. 11, no. 4, pp. 691–719, 2005.

[25] M. Schwager, J. McLurkin, and D. Rus, "Distributed coverage control with sensory feedback for networked robots," in *Robotics: Science and Systems*, 2006, pp. 49–56.

[26] Q. Du, M. Emelianenko, and L. Ju, "Convergence of the lloyd algorithm for computing centroidal voronoi tessellations," *SIAM Journal on Numerical Analysis*, vol. 44, no. 1, pp. 102–119, 2006.

[27] Q. Du, V. Faber, and M. Gunzburger, "Centroidal voronoi tessellations: Applications and algorithms," *SIAM review*, vol. 41, no. 4, pp. 637–676, 1999.

[28] N. Hayashi, K. Segawa, and S. Takai, "2d voronoi coverage control with gaussian density functions by line integration," *SICE Journal of Control, Measurement, and System Integration*, vol. 10, no. 2, pp. 110–116, 2017.

[29] T. Lee, M. Leok, and N. H. McClamroch, "Geometric tracking control of a quadrotor uav on se(3)," in *49th IEEE Conference on Decision and Control (CDC)*. IEEE, 2010, pp. 5420–5425.

[30] F. Bullo and A. D. Lewis, *Geometric control of mechanical systems: modeling, analysis, and design for simple mechanical control systems*. Springer, 2019, vol. 49.

[31] S. P. Bhat and D. S. Bernstein, "A topological obstruction to continuous global stabilization of rotational motion and the unwinding phenomenon," *Systems & control letters*, vol. 39, no. 1, pp. 63–70, 2000.

[32] A. Tayebi and S. McGilvray, "Attitude stabilization of a vtol quadrotor aircraft," *IEEE Transactions on control systems technology*, vol. 14, no. 3, pp. 562–571, 2006.

[33] L. Bartolomei, L. Teixeira, and M. Chli, "Fast multi-uav decentralized exploration of forests," *IEEE Robotics and Automation Letters*, 2023.

[34] K. Guruprasad and P. Dasgupta, "Distributed voronoi partitioning for multi-robot systems with limited range sensors," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 3546–3552.

[35] C. He, Z. Feng, and Z. Ren, "Distributed algorithm for voronoi partition of wireless sensor networks with a limited sensing range," *Sensors*, vol. 18, no. 2, p. 446, 2018.

[36] F. Bullo, J. Cortés, and S. Martinez, *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*. Princeton University Press, 2009.

[37] K. Laventall and J. Cortés, "Coverage control by multi-robot networks with limited-range anisotropic sensory," *International Journal of Control*, vol. 82, no. 6, pp. 1113–1121, 2009.

[38] Z. Zhang, X. Xu, J. Cui, and W. Meng, "Multi-uav area coverage based on relative localization: algorithms and optimal uav placement," *Sensors*, vol. 21, no. 7, p. 2400, 2021.

[39] M. Kegeleirs, D. Garzón Ramos, and M. Birattari, "Random walk exploration for swarm mapping," in *Annual Conference Towards Autonomous Robotic Systems*. Springer, 2019, pp. 211–222.

[40] E. Teruel, R. Aragues, and G. López-Nicolás, "A practical method to cover evenly a dynamic region with a swarm," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1359–1366, 2021.

[41] M. Cao and C. Hadjicostis, "Distributed algorithms for voronoi diagrams and application in ad-hoc networks," *UIUC Coordinated Science Laboratory, Tech. Rep. UILU-ENG-03-2222, DC-210*, 2003.

[42] Y. Song, S. Naji, E. Kaufmann, A. Loquercio, and D. Scaramuzza, "Flightmare: A flexible quadrotor simulator," in *Conference on Robot Learning*, 2020.

[43] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot operating system 2: Design, architecture, and uses in the wild," *Science robotics*, vol. 7, no. 66, p. eabm6074, 2022.

[44] M. Faessler, A. Franchi, and D. Scaramuzza, "Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 620–626, Apr. 2018.

[45] T. Baca, M. Petrlik, M. Vrba, V. Spurny, R. Penicka, D. Hert, and M. Saska, "The mrs uav system: pushing the frontiers of reproducible research, real-world deployment, and education with autonomous unmanned aerial vehicles," *Journal of Intelligent & Robotic Systems*, vol. 102, no. 1, pp. 1–28, 2021.

[46] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial tessellations: concepts and applications of Voronoi diagrams*. John Wiley & Sons, 2009.

[47] F. Pratissoli, B. Capelli, and L. Sabattini, "On coverage control for limited range multi-robot systems," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 9957–9963.

[48] L. C. Pimenta, M. Schwager, Q. Lindsey, V. Kumar, D. Rus, R. C. Mesquita, and G. A. Pereira, "Simultaneous coverage and tracking (scat) of moving targets with robot networks," in *Algorithmic Foundation of Robotics VIII: Selected Contributions of the Eight International Workshop on the Algorithmic Foundations of Robotics*. Springer, 2009, pp. 85–99.

[49] E. Teruel, R. Aragues, and G. López-Nicolás, "A distributed robot swarm control for dynamic region coverage," *Robotics and Autonomous Systems*, vol. 119, pp. 51–63, 2019.

[50] M. Schwager, M. P. Vitus, S. Powers, D. Rus, and C. J. Tomlin, "Robust adaptive coverage control for robotic sensor networks," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 3, pp. 462–476, 2015.

[51] D. Albani, T. Manoni, M. Saska, and E. Ferrante, "Distributed three dimensional flocking of autonomous drones," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 6904–6911.

[52] T. Manoni, D. Albani, J. Horyna, P. Petracek, M. Saska, and E. Ferrante, "Adaptive arbitration of aerial swarm interactions through a gaussian kernel for coherent group motion," *Frontiers in Robotics and AI*, vol. 9, p. 1006786, 2022.

[53] H. Yu and B. M. Wilamowski, "Levenberg–marquardt training," in *Intelligent systems*. CRC Press, 2018, pp. 12–1.

[54] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[55] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[56] B. Zhou, H. Xu, and S. Shen, "Racer: Rapid collaborative exploration with a decentralized multi-uav system," *IEEE Transactions on Robotics*, vol. 39, no. 3, pp. 1816–1835, 2023.

[57] C. Robin and S. Lacroix, "Multi-robot target detection and tracking: taxonomy and survey," *Autonomous Robots*, vol. 40, pp. 729–760, 2016.

[58] J. Gu, T. Su, Q. Wang, X. Du, and M. Guizani, "Multiple moving targets surveillance based on a cooperative network for multi-uav," *IEEE Communications Magazine*, vol. 56, no. 4, pp. 82–89, 2018.

[59] Z. Zhang, Y. Han, Y. Zhou, and M. Dai, "A novel absolute localization estimation of a target with monocular vision," *Optik*, vol. 124, no. 12, pp. 1218–1223, 2013.

[60] B.-F. Wu, W.-C. Lu, and C.-L. Jen, "Monocular vision-based robot localization and target tracking," *Journal of Robotics*, vol. 2011, 2011.

[61] Y. Ming, X. Meng, C. Fan, and H. Yu, "Deep learning for monocular depth estimation: A review," *Neurocomputing*, vol. 438, pp. 14–33, 2021.

[62] J. Chen and D. M. Dawson, "Uav tracking with a monocular camera," in *Proceedings of the 45th IEEE Conference on Decision and Control*. IEEE, 2006, pp. 3873–3878.

[63] A. Ahmad and P. U. Lima, "Multi-robot cooperative object tracking based on particle filters," in *European Conference on Mobile Robots (ECMR)*, 2011, pp. 37–42.

[64] A. Basit, M. N. Dailey, J. Moonrinta, and P. Laksanacharoen, "Joint localization and target tracking with a monocular camera," *Robotics and Autonomous Systems*, vol. 74, pp. 1–14, 2015.

[65] R. Tallamraju, S. Rajappa, M. J. Black, K. Karlapalem, and A. Ahmad, "Decentralized mpc based obstacle avoidance for multi-robot target tracking scenarios," in *2018 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2018, pp. 1–8.

[66] L. Zhou, V. D. Sharma, Q. Li, A. Prorok, A. Ribeiro, P. Tokekar, and V. Kumar, "Graph neural networks for decentralized multi-robot target tracking," in *2022 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2022, pp. 195–202.

[67] J. Alonso-Mora, E. Montijano, T. Nägeli, O. Hilliges, M. Schwager, and D. Rus, "Distributed multi-robot formation control in dynamic environments," *Autonomous Robots*, vol. 43, pp. 1079–1100, 2019.

[68] J. Alonso-Mora, S. Baker, and D. Rus, "Multi-robot formation control and object transport in dynamic environments via constrained optimization," *The International Journal of Robotics Research*, vol. 36, no. 9, pp. 1000–1021, 2017.

[69] L. Sabattini, C. Secchi, and C. Fantuzzi, "Arbitrarily shaped formations of mobile robots: artificial potential fields and coordinate transformation," *Autonomous Robots*, vol. 30, pp. 385–397, 2011.

[70] C. Bai, P. Yan, W. Pan, and J. Guo, "Learning-based multi-robot formation control with obstacle avoidance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 8, pp. 11 811–11 822, 2021.

[71] L. Martins, C. Cardeira, and P. Oliveira, "Inner-outer feedback linearization for quadrotor control: two-step design and validation," *Nonlinear Dynamics*, vol. 110, no. 1, pp. 479–495, 2022.

[72] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 3420–3431.

[73] H. K. Khalil, *Control of Nonlinear Systems*. Prentice Hall, 2002.

[74] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, *Robot Operating System (ROS): The Complete Reference (Volume 1)*. Cham: Springer International Publishing, 2016, ch. RotorS—A Modular Gazebo MAV Simulator Framework, pp. 595–625.

[75] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "Ros: an open-source robot operating system," in *ICRA Workshop on Open Source Software*, vol. 3, no. 3.2, Kobe, Japan, 2009, p. 5.

[76] Aviation Week Editorial Staff, "Korean air raises the bar on drone-based aircraft inspections," https://aviationweek.com/mro/emerging-technologies/korean-air-raises-bar-drone-based-aircraft-inspections, 2023, accessed: 2025-02-28.

[77] J.-D. Boissonnat, M. Sharir, B. Tagansky, and M. Yvinec, "Voronoi diagrams in higher dimensions under certain polyhedral distance functions," in *Proceedings of the eleventh annual symposium on Computational geometry*, 1995, pp. 79–88.

[78] X. Dang, C. Shao, and Z. Hao, "Target detection coverage algorithm based on 3d-voronoi partition for three-dimensional wireless sensor networks," *Mobile Information Systems*, vol. 2019, no. 1, p. 7542324, 2019.

[79] A. Breitenmoser, J.-C. Metzger, R. Siegwart, and D. Rus, "Distributed coverage control on surfaces in 3d space," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 5569–5576.

[80] L. E. Parker, "Multi-robot team design for real-world applications," pp. 91–102, 1996.

[81] Q. Galvane, C. Lino, M. Christie, J. Fleureau, F. Servant, F.-L. Tariolle, and P. Guillotel, "Directing cinematographic drones," *ACM Transactions on Graphics (TOG)*, vol. 37, no. 3, pp. 1–18, 2018.

[82] R. Tallamraju, E. Price, R. Ludwig, K. Karlapalem, H. H. Bülthoff, M. J. Black, and A. Ahmad, "Active perception based formation control for multiple aerial vehicles," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4491–4498, 2019.

[83] A. Bucker, R. Bonatti, and S. Scherer, "Do you see what i see? coordinating multiple aerial cameras for robot cinematography," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 7972–7979.

[84] X. Xu, G. Shi, P. Tokekar, and Y. Diaz-Mercado, "Interactive multi-robot aerial cinematography through hemispherical manifold coverage," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 11 528–11 534.

[85] F. Pratissoli, B. Capelli, and L. Sabattini, "On coverage control for limited range multi-robot systems," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022.

[86] S. Halder and K. Afsari, "Robots in inspection and monitoring of buildings and infrastructure: A systematic review," *Applied Sciences*, vol. 13, no. 4, p. 2304, 2023.

[87] A. Alcántara, J. Capitán, A. Torres-González, R. Cunha, and A. Ollero, "Autonomous execution of cinematographic shots with multiple drones," *IEEE Access*, vol. 8, pp. 201 300–201 316, 2020.

[88] B. Boots, A. Okabe, and K. Sugihara, "Spatial tessellations," *Geographical information systems*, vol. 1, pp. 503–526, 1999.

[89] X. Zheng, R. Ennis, G. P. Richards, and P. Palffy-Muhoray, "A plane sweep algorithm for the voronoi tessellation of the sphere." *Electronic-Liquid Crystal Communications (e-LC)*, 2011.

[90] A. A. Toda, "Radii of the inscribed and escribed spheres of a simplex," *Int. J. Geom*, vol. 3, pp. 5–13, 2014.

[91] M. Hussain, "Yolo-v1 to yolo-v8, the rise of yolo and its complementary nature toward digital manufacturing and industrial defect detection," *Machines*, vol. 11, no. 7, p. 677, 2023.

[92] L. Meier, D. Honegger, and M. Pollefeys, "Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2015, pp. 6235–6240.

[93] J. A. O'Keefe, "The universal transverse mercator grid and projection," *The Professional Geographer*, vol. 4, no. 5, pp. 19–24, 1952.