

2023

# Assessing the Performance of a Particle Swarm Optimization Mobility Algorithm in a Hybrid Wi-Fi/LoRa Flying Ad Hoc Network

William David Paredes

University of North Florida, davidparedesm@gmail.com

Follow this and additional works at: <https://digitalcommons.unf.edu/etd>

Part of the [Aeronautical Vehicles Commons](#), [Artificial Intelligence and Robotics Commons](#), [Digital Communications and Networking Commons](#), [Electrical and Electronics Commons](#), [Multi-Vehicle Systems and Air Traffic Control Commons](#), [Navigation, Guidance, Control and Dynamics Commons](#), [Numerical Analysis and Scientific Computing Commons](#), [Programming Languages and Compilers Commons](#), [Systems and Communications Commons](#), [Systems Engineering and Multidisciplinary Design Optimization Commons](#), and the [Theory and Algorithms Commons](#)

## Suggested Citation

Paredes, William David, "Assessing the Performance of a Particle Swarm Optimization Mobility Algorithm in a Hybrid Wi-Fi/LoRa Flying Ad Hoc Network" (2023). *UNF Graduate Theses and Dissertations*. 1191. <https://digitalcommons.unf.edu/etd/1191>

This Master's Thesis is brought to you for free and open access by the Student Scholarship at UNF Digital Commons. It has been accepted for inclusion in UNF Graduate Theses and Dissertations by an authorized administrator of UNF Digital Commons. For more information, please contact [Digital Projects](#).  
© 2023 All Rights Reserved

# Assessing the Performance of a Particle Swarm Optimization Mobility Algorithm in a Hybrid Wi-Fi/LoRa Flying Ad Hoc Network

by

William David Paredes Molina

A thesis submitted to the School of Engineering in partial fulfillment of the requirements for the  
degree of  
Master of Science in Electrical Engineering

University of North Florida  
College of Computing, Engineering, and Construction

July 2023

## ACKNOWLEDGMENTS

First, I would like to thank my wife Ana and my sons, David and Daniel, for their love and encouragement throughout this challenging journey. Their patience and understanding have been invaluable during times of intense focus and dedication. I am immensely fortunate to have such a loving family by my side, and I am forever grateful for the joy, strength, and inspiration they bring to my life.

I would also like to thank my parents, Norma and Gonzalo, my sisters, Norma and Stefany, and my brother-in-law, Víctor. This achievement would not have been possible without their support and motivation.

I am deeply grateful to my thesis advisor, Dr. Hemani Kaushal, for the guidance and expertise that shaped this research. Her unwavering dedication and insightful feedback enhanced the quality of my thesis. I would also like to thank my co-advisor, Dr. Iman Vakilinia, for his availability and empathetic support, which helped me overcome challenges with confidence. Additionally, I would like to convey my heartfelt thanks to Dr. Zornitza Prodanoff. Besides her ability to balance theoretical frameworks with real-world applications, her pragmatic approach has been invaluable in ensuring the relevance of my study. I feel privileged to have worked under their mentorship, and I am indebted to them for their patience and encouragement.

I am equally thankful to Alethia Wilkerson for going beyond her responsibilities to provide assistance and support. Her outstanding dedication, both professionally and personally, has made this experience more enriching and fulfilling.

Finally, I would like to acknowledge the faculty members and staff of the University of North Florida, including Dr. Christopher Baynard, Dr. Patrick Kreidl, Dr. Alan Harris, Dr. Mona Nasseri, Michael Boyles, Brigid Fitzpatrick, and the late Dr. Chiu Choi, for fostering an enriching environment that facilitated my academic development and personal growth.

# Contents

|  |             |
|--|-------------|
| <b>ACKNOWLEDGMENTS .....</b>                             | <b>ii</b>   |
| <b>Contents .....</b>                                    | <b>iii</b>  |
| <b>List of Tables .....</b>                              | <b>vii</b>  |
| <b>List of Figures.....</b>                              | <b>ix</b>   |
| <b>ABSTRACT.....</b>                                     | <b>xiii</b> |
| <b>1 Introduction .....</b>                              | <b>1</b>    |
| 1.1 Motivation .....                                     | 5           |
| 1.2 Objectives and Contributions .....                   | 5           |
| 1.3 Organization .....                                   | 6           |
| <b>2 Technical Overview .....</b>                        | <b>7</b>    |
| 2.1 UAV Taxonomy .....                                   | 8           |
| 2.2 Differences between FANETs, VANETs, and MANETs ..... | 9           |
| 2.3 FANET Communications.....                            | 10          |
| 2.3.1 Network Architecture.....                          | 11          |
| 2.3.2 Communication Channel .....                        | 13          |
| 2.3.3 Low Power Wide Area Networks (LPWANs).....         | 18          |
| 2.4 LoRa .....   | 19          |
| 2.4.1 CSS Modulation.....                                | 19          |
| 2.4.2 Frequency.....                                     | 20          |
| 2.4.3 Bandwidth (BW) .....                               | 20          |
| 2.4.4 Spreading Factor (SF) .....                        | 21          |
| 2.4.5 Coding Rate (CR) Index .....                       | 21          |
| 2.4.6 Transmission Power.....                            | 22          |
| 2.4.7 LoRa Frame Format .....                            | 23          |
| 2.5 IEEE 802.11 .....                                    | 24          |
| 2.5.1 IEEE 802.11p.....                                  | 25          |

|          |   |           |
|----------|---|-----------|
| 2.5.2    | IEEE 802.11s .....                                      | 25        |
| 2.5.3    | IEEE 802.11ax .....                                     | 25        |
| 2.6      | FANET Mobility .....                                    | 26        |
| 2.6.1    | Mobility objectives .....                               | 27        |
| 2.6.2    | Mobility models .....                                   | 27        |
| 2.7      | Optimization Approach to the Mobility Problem .....     | 29        |
| 2.7.1    | Global and Local Optimization.....                      | 30        |
| 2.7.2    | Single-Objective and Multiobjective Optimization .....  | 31        |
| 2.7.3    | Single-Solution and Multiple-Solution Optimization..... | 31        |
| 2.7.4    | Particle Swarm Optimization.....                        | 32        |
| 2.8      | Related work .....                                      | 33        |
| 2.8.1    | FANET Architectures Involving LoRa or LoRaWAN.....      | 33        |
| 2.8.2    | FANET Mobility.....                                     | 33        |
| <b>3</b> | <b>Simulation Framework Development.....</b>            | <b>36</b> |
| 3.1      | System Architecture .....                               | 37        |
| 3.2      | Considerations and Assumptions .....                    | 38        |
| 3.2.1    | Communications .....                                    | 38        |
| 3.3      | Definition of the Objective Function .....              | 40        |
| 3.3.1    | Multiobjective Optimization.....                        | 45        |
| 3.3.2    | Problem Formulation .....                               | 49        |
| 3.4      | Solution Using a PSO Mobility Model .....               | 50        |
| 3.4.1    | Search Space Constraint Handling.....                   | 51        |
| 3.4.2    | Kinematic Constraints Handling.....                     | 52        |
| 3.4.3    | Dynamic Clustering .....                                | 55        |
| 3.4.4    | Hybrid PSO.....   | 57        |
| 3.4.5    | Stopping Criteria.....                                  | 60        |
| 3.5      | Simulation Parameters.....                              | 61        |
| 3.5.1    | Node configuration .....                                | 61        |
| 3.5.2    | Propagation Models .....                                | 63        |
| 3.5.3    | Mobility.....   | 64        |
| <b>4</b> | <b>Results and Analysis.....</b>                        | <b>68</b> |
| 4.1      | Comparison between Different Objective Functions .....  | 68        |
| 4.1.1    | Coverage Reward Only .....                              | 70        |

|          |  |            |
|----------|--|------------|
| 4.1.2    | Coverage and Path Maintenance Rewards.....                                   | 73         |
| 4.2      | Comparison between Different PSO Configurations .....                        | 85         |
| 4.2.1    | Adaptive Inertia Weight vs. Guaranteed Convergence Parameters.....           | 86         |
| 4.2.2    | Propagation Model Alternatives .....   | 89         |
| 4.2.3    | Stopping Criteria Alternatives .....   | 92         |
| 4.3      | Performance Metrics .....  | 95         |
| 4.3.1    | Number of Iterations Required for Stabilization or Stoppage ( $ts$ ).....    | 95         |
| 4.3.2    | Percentage of Covered Ground Nodes ( $C\%$ ).....                            | 95         |
| 4.3.3    | FANET Coverage Efficiency ( $\eta C$ ).....                                  | 96         |
| 4.3.4    | Percentage of Ground Nodes with a Path to the Control Station ( $P\%$ )..... | 97         |
| 4.3.5    | Overall Efficiency ( $\eta O$ ).....   | 97         |
| 4.4      | Performance Evaluation of the Different Mobility Algorithms .....            | 98         |
| 4.4.1    | Number of Iterations .....   | 99         |
| 4.4.2    | Percentage of Covered Ground Nodes.....                                      | 101        |
| 4.4.3    | FANET Coverage Efficiency.....   | 102        |
| 4.4.4    | Percentage of Ground Nodes with a Path to the Control Station.....           | 103        |
| 4.4.5    | Overall Efficiency .....   | 105        |
| <b>5</b> | <b>Conclusions and Future Work .....</b>                                     | <b>107</b> |
| 5.1      | Conclusions .....  | 107        |
| 5.1.1    | Regarding the Objective Function and Multiobjective Optimization.....        | 107        |
| 5.1.2    | Regarding the Optimization Algorithm .....                                   | 109        |
| 5.1.3    | Regarding Overall Performance .....  | 111        |
| 5.1.4    | Regarding the Use of LoRa.....   | 111        |
| 5.2      | Future Work .....  | 112        |
| 5.2.1    | Communications .....   | 112        |
| 5.2.2    | Mobility based on Multiobjective Optimization.....                           | 113        |
| 5.2.3    | Energy .....   | 114        |
| 5.2.4    | The Use of LoRa in FANETs .....  | 114        |
|          | <b>REFERENCES.....</b>   | <b>115</b> |
|          | <b>APPENDIX A: Main MATLAB Script.....</b>                                   | <b>122</b> |
|          | <b>APPENDIX B: Implementation of Dijkstra's Algorithm in MATLAB .....</b>    | <b>142</b> |

|   |            |
|---|------------|
| <b>APPENDIX C: Modified MATLAB Functions to Load Configuration Parameters .....</b>                     | <b>144</b> |
| <b>APPENDIX D: Function Used to Obtain the Path Loss Table.....</b>                                     | <b>148</b> |
| <b>APPENDIX E: Function to Implement the Free Space Propagation Model .....</b>                         | <b>149</b> |
| <b>APPENDIX F: Performance Evaluation of the Different Mobility Algorithms (Tabulated Results).....</b> | <b>151</b> |

# List of Tables

|  |    |
|--|----|
| Table 2.1: UAV taxonomy.....   | 9  |
| Table 2.2: Radio channel models and their characteristics.....   | 16 |
| Table 2.3: Summary of LoRa modulation parameters.....  | 22 |
| Table 2.4: Summary of IEEE 802.11 standards and their characteristics.....   | 24 |
| Table 3.1: MAC and PHY configuration parameters. ....  | 62 |
| Table 3.2: Application traffic configuration. ....   | 62 |
| Table 3.3: General mobility parameters.....  | 64 |
| Table 3.4: PSO configuration parameters. ....  | 66 |
| Table 4.1: Model configuration parameters.....   | 69 |
| Table 4.2: Link and trajectory plot legends for Figures 4.1(a)–4.18(a).....  | 69 |
| Table 4.3: Alternative gains applied only for covering multiple ground nodes using Hybrid PSO (1500 m x 1500 m)..... | 70 |
| Table 4.4: Coverage and path maintenance reward alternatives using Hybrid PSO (1500 m x 1500 m). ....                | 73 |
| Table 4.5: Coverage and path maintenance reward alternatives using Hybrid PSO (2000 m x 2000 m). ....                | 75 |
| Table 4.6: Coverage and path maintenance reward alternatives using Hybrid PSO (2000 m x 2000 m). ....                | 78 |
| Table 4.7: Coverage and path maintenance reward alternatives using PSO-only (1500 m x 1500 m). ....                  | 80 |
| Table 4.8: Coverage and path maintenance reward alternatives using PSO-only (2000 m x 2000 m). ....                  | 83 |



|  |    |
|--|----|
| Table 4.9: Summary of results for the objective function analysis for the Hybrid PSO. ....                   | 85 |
| Table 4.10: Summary of results for the objective function analysis for PSO-only. ....                        | 85 |
| Table 4.11: Adaptive inertia weight vs. guaranteed convergence parameters.....                               | 86 |
| Table 4.12: Propagation model alternatives using adaptive inertia weight. ....                               | 89 |
| Table 4.13: Stopping criteria alternatives using adaptive inertia weight and log-normal<br>propagation. .... | 92 |
| Table 4.14: Objective function and PSO configuration.....  | 98 |
| Table 4.15: Assessment scenarios.....  | 98 |
| Table 4.16: Final model configuration parameters. ....   | 99 |

# List of Figures

|  |    |
|--|----|
| Figure 1.1: Proposed scenario.....   | 4  |
| Figure 2.1: Relationship between MANET, VANET and FANET.....   | 7  |
| Figure 2.2: FANET types of links.....  | 11 |
| Figure 2.3: FANET topologies. ....   | 13 |
| Figure 2.4: Chirp spread spectrum (CSS) modulation.....  | 20 |
| Figure 2.5: LoRa frame format .....  | 23 |
| Figure 3.1: System architecture. ....  | 37 |
| Figure 3.2: Approaches to distance calculation between two points in a square grid of side $l$ . ...   | 43 |
| Figure 3.3: Air-to-ground distance at height $h$ for the expected ground-to-ground distance $d$ . ...  | 44 |
| Figure 3.4: Proximity ranges between UAVs. ....  | 46 |
| Figure 3.5: Basic PSO algorithm flowchart.....   | 51 |
| Figure 3.6: Dynamic clustering.....  | 55 |
| Figure 3.7: Example of traffic configuration for ten ground nodes and a Wi-Fi backhaul. ....   | 63 |
| Figure 4.1: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. .... | 71 |
| Figure 4.2: Topology and convergence results when applying a distance-based gain for covering multiple ground nodes (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. ....     | 72 |
| Figure 4.3: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. .... | 74 |

|  |    |
|--|----|
| Figure 4.4: Topology and convergence results when applying a distance-based gain for having a path to the control station (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. ....                         | 75 |
| Figure 4.5: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (Hybrid PSO 2000m). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.....                      | 76 |
| Figure 4.6: Topology and convergence results when applying a distance-based gain for having a path to the control station (Hybrid PSO 2000m). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. ....                   | 77 |
| Figure 4.7: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (Hybrid PSO 2000m until convergence). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. ....   | 78 |
| Figure 4.8: Topology and convergence results when applying a distance-based gain for having a path to the control station (Hybrid PSO 2000m until convergence). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. .... | 79 |
| Figure 4.9: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (PSO-only). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. ....                             | 81 |
| Figure 4.10: Topology and convergence results when applying a distance-based gain for having a path to the control station (PSO-only). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. ....                          | 82 |

|   |    |
|---|----|
| Figure 4.11: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (PSO-only 2000m). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.....    | 83 |
| Figure 4.12: Topology and convergence results when applying a distance-based gain for having a path to the control station (PSO-only 2000m). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. .... | 84 |
| Figure 4.13: Topology and convergence results when using adaptive inertia weight (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.....  | 87 |
| Figure 4.14: Topology and convergence results when using guaranteed convergence PSO parameters (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.....                                  | 88 |
| Figure 4.15: Topology and convergence results when using a ray tracing propagation model (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. ....                                       | 90 |
| Figure 4.16: Topology and convergence results when using a log-normal propagation model (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. ....  | 91 |
| Figure 4.17: Topology and convergence results when using signal stability as fitness function (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. ....                                  | 93 |

|  |     |
|--|-----|
| Figure 4.18: Topology and convergence results when using position stability as fitness function (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV. .... | 94  |
| Figure 4.19: Number of iterations required for stabilization or stoppage ( $t_s$ ). (a) Ray tracing propagation model. (b) Log-normal propagation model. (c) Log-normal propagation model with LoRa backhaul. ....         | 100 |
| Figure 4.20: Percentage of covered ground nodes ( $C\%$ ). (a) Ray tracing propagation model. (b) Log-normal propagation model. (c) Log-normal propagation model with LoRa backhaul. ....                                  | 101 |
| Figure 4.21: FANET Coverage Efficiency ( $\eta_C$ ). (a) Ray tracing propagation model. (b) Log-normal propagation model. (c) Log-normal propagation model with LoRa backhaul. ....  | 103 |
| Figure 4.22: Percentage of ground nodes with a path to the control station ( $P\%$ ). (a) Ray tracing propagation model. (b) Log-normal propagation model. (c) Log-normal propagation model with LoRa backhaul. ....       | 104 |
| Figure 4.23: Overall efficiency ( $\eta_O$ ). (a) Ray tracing propagation model. (b) Log-normal propagation model. (c) Log-normal propagation model with LoRa backhaul. ....   | 106 |

## ABSTRACT

Research on Flying Ad-Hoc Networks (FANETs) has increased due to the availability of Unmanned Aerial Vehicles (UAVs) and the electronic components that control and connect them. Many applications, such as 3D mapping, construction inspection, or emergency response operations could benefit from an application and adaptation of swarm intelligence-based deployments of multiple UAVs. Such groups of cooperating UAVs, through the use of local rules, could be seen as network nodes establishing an ad-hoc network for communication purposes.

One FANET application is to provide communication coverage over an area where communication infrastructure is unavailable. A crucial part of a FANET implementation is computing the optimal position of UAVs to provide connectivity with ground nodes while maximizing geographic span. To achieve optimal positioning of FANET nodes, an adaptation of the Particle Swarm Optimization (PSO) algorithm is proposed. A 3D mobility model is defined by adapting the original PSO algorithm and combining it with a fixed-trajectory initial flight. A Long Range (LoRa) mesh network is used for air-to-air communication, while a Wi-Fi network provides air-to-ground communication to several ground nodes with unknown positions. The optimization problem has two objectives: maximizing coverage to ground nodes and maintaining an end-to-end communication path to a control station, through the UAV mesh. The results show that the hybrid mobility approach performs similarly to the fixed trajectory flight regarding coverage, and outperforms fixed trajectory and PSO-only algorithms in both path maintenance and overall network efficiency, while using fewer UAVs.

# **Chapter 1**

## **Introduction**

The past few decades have witnessed the “wireless revolution” [1,2], a significant increase in the number of connected devices. These devices, such as body sensors, light dimmers, vacuum cleaners, thermostats, refrigerators, and autonomous vehicles, have been designed to simplify our lives both at home and in the workplace. This expansion has been so substantial that numerous sources now indicate that the number of connected devices has far exceeded the number of connected individuals, reaching an estimated count of tens of billions [3-5]. For these interconnected devices to function effectively, they require the ability to communicate with each other and with users. This concept is widely referred to as the Internet of Things (IoT). In essence, IoT encompasses a collection of objects equipped with sensors and actuators that are linked together via either private or public networks, with the Internet being the most common option. The growth of the IoT, like other technological advancements, has been motivated by the aspiration to enhance quality of life in a cost-efficient manner [3].

While human communications often demand a considerable bandwidth (e.g., on-demand video streaming and file sharing) and have little tolerance to delay (e.g., voice calls and videoconference), communications between objects have different characteristics, at least for the time being. IoT communications usually involve low data rates, where a relaxation of bandwidth constraints usually results in lower power consumption and a longer communication range, considering that these devices might be placed at isolated locations with no access to a power grid or Wide Area Networks (WANs) such as those available through internet service providers, mobile carriers, or proprietary deployments.

In a natural step in the development of wireless technologies, Ad Hoc networks were conceived as a method of connecting nodes when no centralized infrastructure is available, and the concept of mobility was immediately tied to this type of network [6]. Mobile Ad-Hoc Networks (MANETs) have been used for military applications for many years [7]. As they increasingly connect more and more objects across the globe, MANETs have become one of the fundamental network paradigms in IoT. Moreover, the recent surge in the availability of Unmanned Aerial Vehicles (UAVs), also known as drones, has given rise to the concept of Flying Ad-Hoc Networks (FANETs). FANETs are a type of wireless network that consists of UAVs that communicate with each other to provide wireless connectivity.

In a FANET, UAVs act as nodes that communicate with each other to form a network. These nodes can either act as routers or end devices, and they use wireless communication protocols to exchange data. The network topology in FANETs is dynamic, as UAVs move around in different directions and distances, and the network must adapt to the changes in topology.

The use of UAVs in FANETs provides several advantages over traditional wireless networks. UAVs can be deployed in areas where traditional networks are not available or are destroyed, such as disaster-stricken areas, remote locations, or war zones. Additionally, UAVs can be rapidly deployed and redeployed to cover a larger area or to provide connectivity to a specific location. Furthermore, FANETs can provide a higher degree of reliability and fault tolerance, as UAVs can act as relays and reconfigure the network in case of a node failure or damage.

Despite the potential advantages of FANETs, there are several challenges that need to be addressed to make them a practical solution. One of the main challenges is the design of communication protocols that can operate in a highly dynamic and unpredictable environment. The communication protocols should be able to handle the rapid changes in topology, bandwidth, latency, and



interference, and should be scalable to support a large number of nodes. The communication protocols should also be energy-efficient, as the UAVs have limited battery life and need to conserve energy for flight and other tasks.

Another challenge in FANETs is the development of robust mobility algorithms for autonomous flight that are capable of maintaining efficient and reliable communication among UAVs, adapting to environmental disturbances and changes in the network, while minimizing energy consumption and avoiding collisions.

The design of FANETs should consider the regulatory and ethical issues related to the use of UAVs. The use of UAVs for commercial or civilian purposes is subject to regulations and guidelines set by the civil aviation authorities, such as the Federal Aviation Administration (FAA) in the United States or the European Aviation Safety Agency (EASA) in Europe. These regulations cover various aspects of UAV operations, such as airworthiness, pilot certification, flight restrictions, and privacy.

While existing wireless technologies —such as cellular and Wi-Fi— have been adopted for IoT [8,9], Low Power Wide Area Networks (LPWANs) have been developed specifically as one of IoT’s enabling technologies for long-range applications [10-13]. LoRa (named after “long range”) is a physical-layer (PHY) LPWAN technology that provides long-range communication at low data rates. Due to its scalability, low power consumption, and ease of deployment, LoRa technology has recently gained significant attention from researchers recently. Its attributes make it suitable for IoT applications, particularly when used as part of LoRaWAN —a protocol used to create a star topology network using LoRa technology. However, when it comes to MANETs and FANETs, LoRaWAN presents some limitations regarding its star topology, its medium access control (MAC) layer and its lack of routing procedures [14]. Some work has been done to assess

the performance of LoRa without the constraints of LoRaWAN, for static and ground-mobile ad-hoc mesh networks in typical IoT scenarios [15-23]. Nevertheless, there is little research activity on FANETs using LoRa technology.

In this research, an optimization approach is adopted to govern FANET mobility and to ultimately determine UAV positions that maximize network coverage to a series of ground nodes. The proposed FANET, shown in Figure 1.1, relies on a Wi-Fi access network (air-to-ground links), and a backhaul mesh network (air-to-air links) consisting of LoRa or Wi-Fi, depending on the application. An adaptation of Particle Swarm Optimization (PSO) is proposed as the optimization algorithm used to achieve UAV mobility and self-positioning.

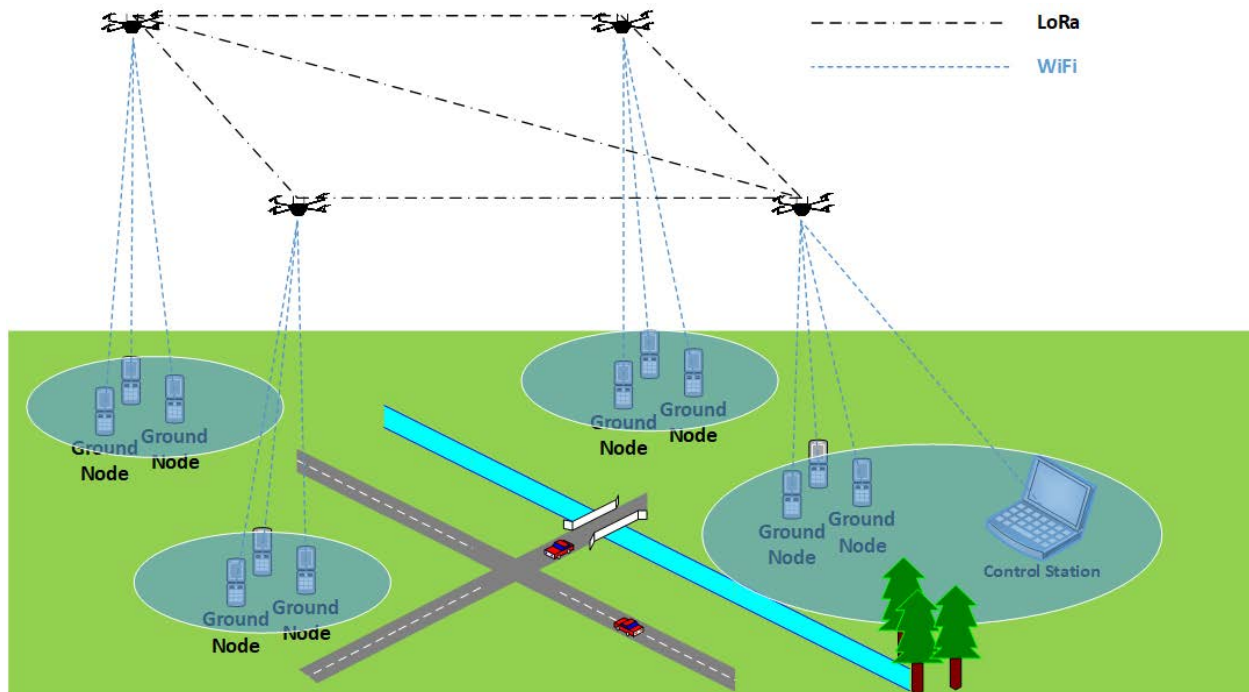


Figure 1.1: Proposed scenario.

## 1.1 Motivation

The motivation behind conducting research on FANET mobility using PSO and LoRa lies in the pursuit of efficient and reliable communication in aerial networks. FANETs have emerged as a promising solution for various applications, including disaster management, environmental monitoring, and surveillance. However, ensuring seamless communication and optimal mobility of UAVs in dynamic environments remains a significant challenge. By integrating PSO, a nature-inspired optimization algorithm, with LoRa, a low-power, long-range wireless technology, the aim is to develop an approach to enhance FANET mobility. PSO can be utilized to optimize UAV movement to maximize network coverage or minimize energy consumption. Additionally, leveraging the long-range capabilities of LoRa enables reliable and long-distance communication among UAVs, facilitating data exchange and coordination. The use of Wi-Fi as access network accounts for the widespread availability of these devices among end users.

## 1.2 Objectives and Contributions

The main goal of this research is to develop and assess the performance of a FANET that maximizes network coverage to ground nodes with unknown positions. To achieve this goal, the following contribution have been made:

- An exploration of the state of the art of FANET mobility.
- An exploration of the state of the art of the use of LoRa technology in FANETs.
- The development of a comprehensive FANET simulation framework.
- The definition of a single objective function that contains the elements to achieve multiobjective optimization.
- The development of a PSO algorithm that governs autonomous flight, while taking into consideration a wide range of constraints associated with the proposed scenario.

- The definition of performance metrics to measure the fulfilment of multiple optimization objectives separately and jointly.
- The performance evaluation of the proposed FANET in terms of the defined performance metrics.

## **1.3 Organization**

The remainder of this thesis is organized as follows: Chapter 2 presents a technical overview of FANETs, LoRa, Wi-Fi, and PSO, as well as a summary of related work. Chapter 3 covers the aspects involved in the development of the simulation framework. The presentation of the results and a discussion of the findings are contained in Chapter 4. Finally, the concluding remarks and future work are stated in Chapter 5.

# Chapter 2

## Technical Overview

In recent years, there has been an increasing interest in the development of Flying Ad Hoc Networks (FANETs). FANETs are a type of Mobile Ad Hoc Network (MANET) that consists of unmanned aerial vehicles (UAVs) or drones that communicate with each other wirelessly to perform various tasks such as surveillance, reconnaissance, search and rescue, environmental monitoring, and communication relay. More specifically, FANETs are also a subset of Vehicular Ad Hoc Networks (VANETs). These networks are flexible, versatile, and have the potential to operate in various challenging environments in which it is difficult or impossible for other types of networks to operate. Figure 2.1 shows the relationship between MANETs, VANETs, and FANETs [24].

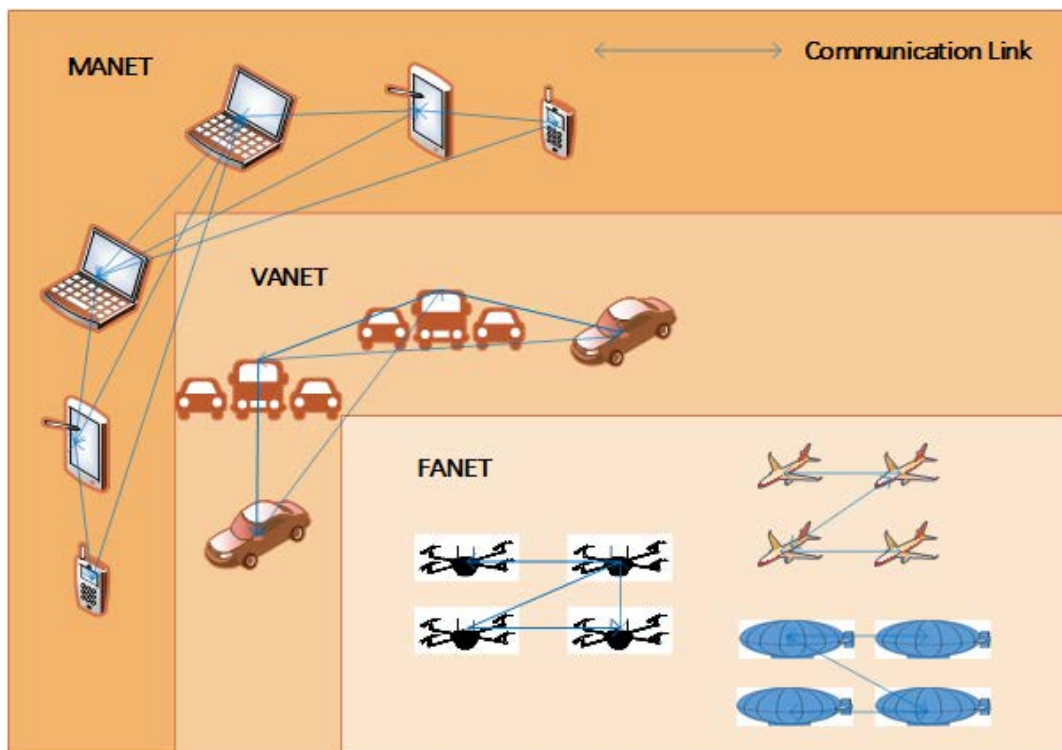


Figure 2.1: Relationship between MANET, VANET and FANET.

FANETs are typically self-organizing, decentralized networks, which do not require any pre-existing infrastructure. Each UAV in the network acts as a node, and they communicate with each other using wireless links. These networks are highly dynamic, and the topology changes rapidly as the UAVs move around, making it challenging to maintain reliable communication and coordination among them.

The technical topics involved in FANETs include:

- **Wireless communication:** FANETs rely on wireless communication technologies, such as IEEE 802.11 (Wi-Fi), Bluetooth, LoRa, Zigbee, combined with built-in or supplementary ad hoc networking protocols, to enable communication between UAVs. These wireless technologies allow UAVs to exchange data, such as sensor readings, location information, and control commands, without the need for a physical wired connection.
- **Mobility:** Mobility is a critical aspect of FANETs, as it directly affects network performance, energy consumption, and safety. Mobility models describe the movement patterns of UAVs in FANETs. These models are used to simulate the behavior of UAVs in different scenarios and evaluate the performance of FANET protocols and algorithms.
- **Energy:** FANETs have limited battery life, and power management is critical to ensure that UAVs can operate for an extended period without requiring frequent battery replacement. Power management techniques include duty cycling, sleep mode, and energy harvesting.

## **2.1 UAV Taxonomy**

Regarding UAV taxonomy, multiple categorizations are proposed in [25-27]. A summary of this topic is presented in Table 2.1.

Table 2.1: UAV taxonomy.

| By wing type        |  |  | By size                        | By type of flight                           | By flight range  |                                      |                                      |  | By energy autonomy |  |  | By altitude |  | By purpose |  |  |  |  |  |  |
|---------------------|--|--|--------------------------------|---|--|--------------------------------------|--------------------------------------|--|--------------------|--|--|-------------|--|------------|--|--|--|--|--|--|
| Fixed wing [25-27]  |  |  | Large [25,26]<br>Small [25,26] | Autonomous [26]<br>Remotely controlled [26] | Close-range [25]<br>Short-range [25]<br>Mid-range [25]<br>Long-range | High [26]<br>Medium [26]<br>Low [26] | High [26]<br>Medium [26]<br>Low [26] | Military [27]<br>Communications [26,27]<br>Surveillance [27]<br>Photography/Mapping [25,27]<br>Exploration/Surveying [27]<br>Remote sensing [27]<br>Delivery [25]<br>First-Person View |                    |  |  |             |  |            |  |  |  |  |  |  |
| Rotary wing [25-27] |  |  |                                |   |  |                                      |                                      |  |                    |  |  |             |  |            |  |  |  |  |  |  |
| Hybrid [27]         |  |  |                                |   |  |                                      |                                      |  |                    |  |  |             |  |            |  |  |  |  |  |  |

## 2.2 Differences between FANETs, VANETs, and MANETs

The differences between FANETs, VANETs, and MANETs are analyzed from different perspectives in [25-29]. The aforementioned works coincide with the fact that FANETs have specific characteristics. Those specificities are summarized in the following fields:

- Node mobility: Contrary to the elements of MANETs and ground VANETs, UAVs experience relatively fewer obstacles, which allows them to move in and around three axes with a certain amount of freedom at somewhat constant speeds. However, holding a fixed position can be more challenging, or even impossible, depending on weather conditions and the type of UAV. These circumstances influence the mobility model to be applied but also impact other characteristics, such as node density, topology change rate, localization alternatives, and applicable propagation models.
- Radio propagation: The presence of fewer obstacles allows for the consideration of mostly line-of-sight (LoS) propagation, especially for air-to-air links, while taking into account

weather conditions and the Doppler effect caused by the speed of UAVs relative to the ground and to one another. Air-to-air and air-to-ground are the two main types of links that can be identified, although air-to-satellite links might also be considered for some applications.

- Energy constraints: They depend on the type of UAV. Battery-powered UAVs are more energy-constrained, making it useful to have communication hardware that consumes less power, allowing for increased flight time, although most of the energy is dedicated to keep the UAV and its payload in the air. Large fixed-wing UAVs are most likely powered by combustion engines that can carry and charge larger batteries, making them less energy-constrained.

## **2.3 FANET Communications**

UAVs communicate with each other using wireless protocols such as IEEE 802.11 (Wi-Fi), IEEE 802.15.4 (which is an IEEE standard for low-rate wireless personal area networks, or LR-WPANs), and Low Power Wide Area Network (LPWAN) protocols, among others. These protocols must be designed to handle the unique characteristics of FANETs, including high mobility, limited bandwidth, and dynamic network topologies.

FANETs require efficient routing protocols that can adapt to the changing network topologies and ensure reliable communication between UAVs. Some of the commonly used routing protocols in FANETs include Ad Hoc On-Demand Distance Vector (AODV), Optimized Link State Routing (OLSR), and Dynamic Source Routing (DSR). FANETs face various security and privacy challenges, such as eavesdropping, spoofing, and denial of service attacks. Security and privacy mechanisms, such as encryption, digital signatures, and access control, must be implemented to protect FANETs from such threats.



### 2.3.1 Network Architecture

The network architecture is highly dependent on the FANET application. Single UAV architectures are not considered as FANETs because FANETs are composed of more than one UAV and communication between UAVs cannot rely on infrastructure networks [28]. Moreover, this means that topologies can be mesh, star-of-meshes, or mesh-of-meshes.

As mentioned in Section 2.2, three different types of links can be identified according to the location of the elements they connect, namely air-to-air (UAV to UAV), air-to-ground (UAV to ground) and air-to-satellite (UAV to satellite). Links can also be classified according to the role they play from a communications network perspective: access links, backhaul links, and backbone links. The two mentioned classifications are illustrated in Figure 2.2.

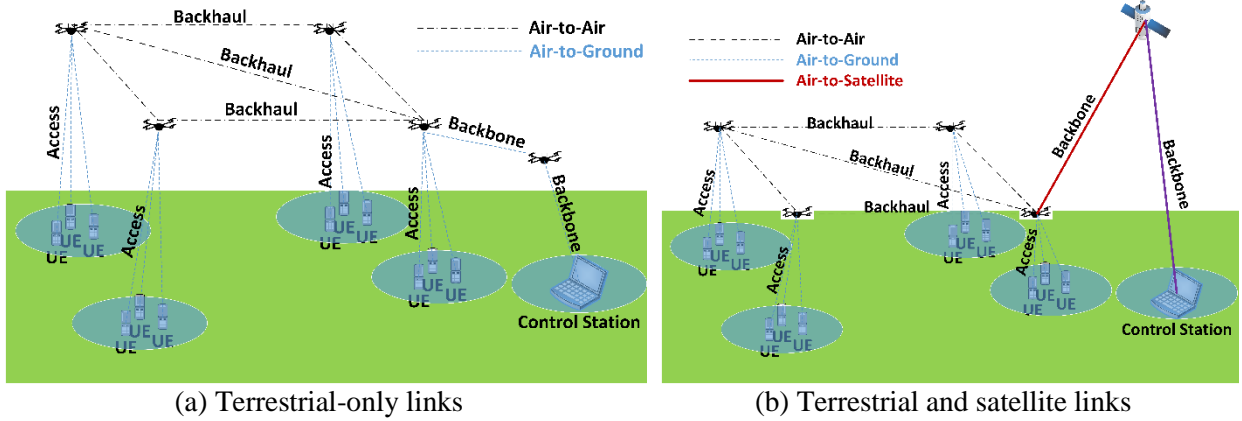


Figure 2.2: FANET types of links.

Four UAV communication architectures are mentioned in [25], based on the type of infrastructure utilized: UAV direct communication, UAV communication via satellite networks, UAV communication via cellular networks, and UAV communication via Ad-Hoc networks. However, these could be further summarized into communication through infrastructure and infrastructure-less communication, hence, only UAV communication via Ad-Hoc networks corresponds to FANETs.

Three hierarchical architectures are described in [30,31], based on how the UAVs connect to each other and to a ground base station. The first architecture relies on a single UAV acting as a hub to connect a single group of UAVs to the base station. The second architecture involves clustering UAVs into groups, each one of them having one hub to connect to the base station. Finally, in the third architecture, multiple layers of UAV clusters connect to each other through one root UAV, and only one of the groups has a hub that connects all others to the base station. According to the description, UAV-to-UAV communication relies on low-power, short-range links, while UAV-to-ground communication does it on high-power, long-range links. These architectures could be employed in applications where compact swarms with longer UAV-to-ground ranges—when compared to UAV-to-UAV ranges—are needed. Moreover, having centralized links to the base station implies one or more of the following situations:

- Most communications would take place inside the swarms.
- Communication with the base station is less frequent or takes place at low data rates.
- The UAV that handles the link to the base station is a single point of failure and may become a bottleneck.

To overcome the last issue, the authors of [32] propose a multi-layer architecture where clusters are grouped in layers and each cluster selects a UAV that acts as a hub to connect to the base station or to another layer, and a second UAV as a backup hub. The UAV clusters can be grouped two-dimensionally or in multiple layers in the three-dimensional space. A similar multi-layer approach is also presented in [33]. The architectures described in [30-33] can be synthesized by topology into the categories shown in Figure 2.3.

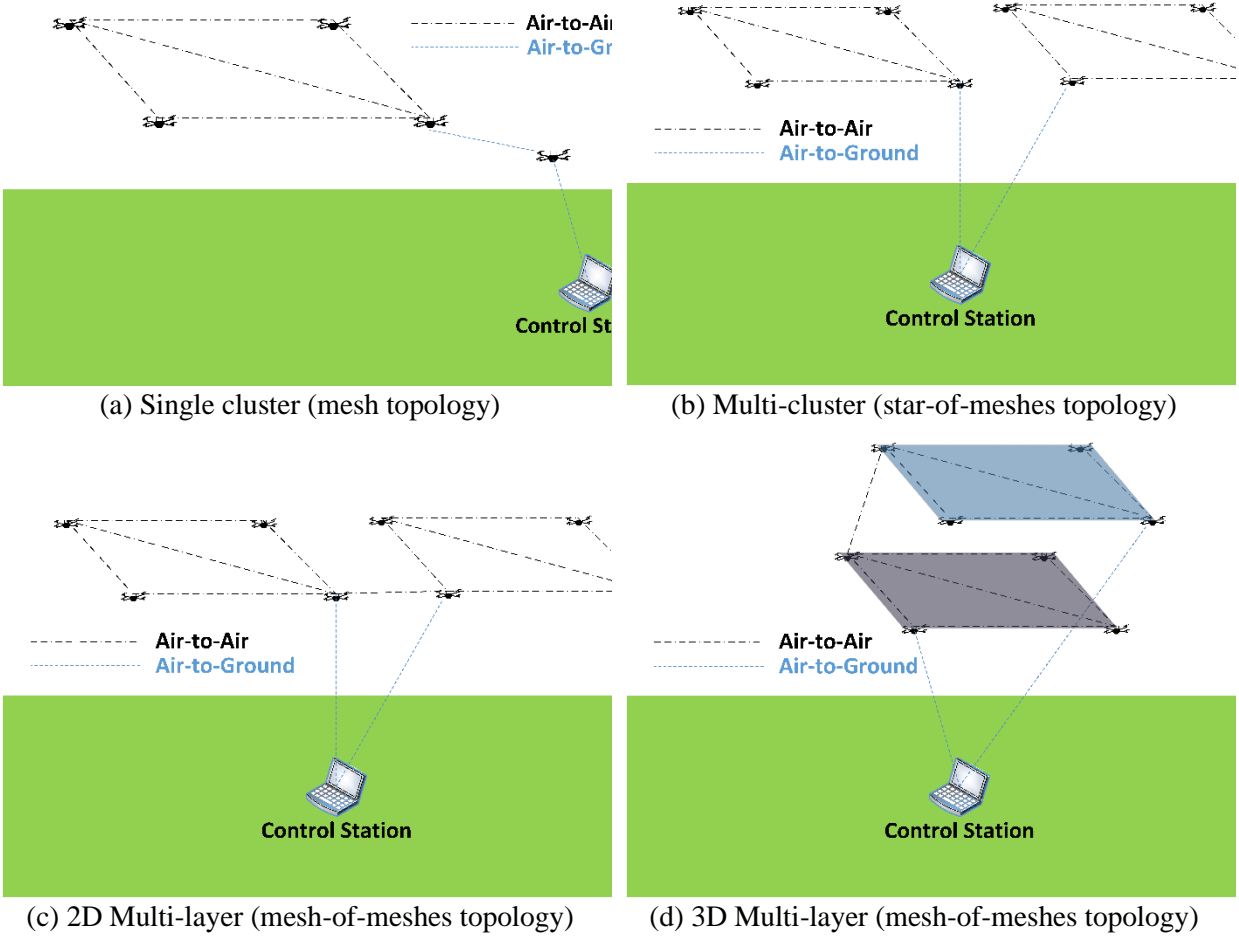


Figure 2.3: FANET topologies.

### 2.3.2 Communication Channel

The wireless channel, or radio channel, refers to the wireless communication medium through which signals are transmitted and received in wireless communication systems. The radio channel is a harsh environment that poses restrictions on any wireless communication system. Modeling it is a challenging task that depends on environmental factors, as well as on the specific application and its required level of detail.

### **A. Radio Channel Models**

In general, radio channel models are used to simulate the effects of the wireless channel on a transmitted signal, by taking into account the physical characteristics of the environment, the transmitter and receiver parameters, and the propagation mechanisms involved.

The free space propagation model describes the attenuation or loss of signal strength as electromagnetic waves propagate through free space. In this model, the signal power decreases proportionally to the square of the inverse of the distance from the transmitter. The free space propagation model assumes an ideal scenario without any obstructions, diffraction, or reflection, making it a simplified representation used to estimate the signal strength in open-air environments.

The log-normal propagation model is a statistical model used to describe the variability of signal strength in wireless communication. Unlike the free space propagation model, it considers the effects of obstacles, reflections, and diffraction, which can introduce random fluctuations in signal strength. The model assumes that the received signal strength follows a log-normal distribution, where the logarithm of the signal strength is normally distributed. This distribution captures the variability caused by multipath propagation and environmental factors. The log-normal propagation model is widely used in wireless system design, especially in scenarios where signal strength variations need to be accounted for, such as urban environments with buildings and other obstacles.

Ray tracing can be considered a family of models that share the basic principle of tracing the path of electromagnetic waves as rays through a given environment. In a ray tracing model, the environment is divided into a number of discrete regions, each with its own set of physical properties such as refractive index, absorption coefficient, and reflection coefficient. The model

then traces individual rays through these regions, calculating the angle of reflection or refraction at each interface, as well as the amount of attenuation due to absorption.

A list of channel models and their characteristics is presented in Table 2.2. Note that this is not an exhaustive list of radio channel models, and there may be variations in the equations depending on the specific implementation.

Table 2.2: Radio channel models and their characteristics.

| Model                          | Propagation Effects          | Environment    | Equation / Output-Description  |
|--------------------------------|------------------------------|----------------|--|
| Free space [34]                | Path loss                    | Indoor/Outdoor | $P_r = P_t G_t G_r \left( \frac{\lambda}{4\pi d} \right)^2$ (2.1)  |
| Two-ray ground reflection [34] | Path loss, reflection        | Outdoor        | $P_r = \frac{P_t G_t G_r (h_t h_r)^2}{d^4}$ (2.2)  |
| Log-distance [34]              | Path loss                    | Indoor/Outdoor | $P_r = P_t G_t G_r \left( \frac{\lambda}{4\pi d_0} \right)^2 \left( \frac{d_0}{d} \right)^n$ (2.3)   |
| Log-normal [34]                | Path loss, shadowing         | Indoor/Outdoor | $PL(d)[dB] = PL(d_0) + 10n \log \left( \frac{d}{d_0} \right) + X_\sigma$ (2.4)   |
| Longley-Rice [34]              | Path loss, shadowing         | Outdoor        | The model operates in different modes  |
| Rayleigh [34]                  | Fading                       | Outdoor [35]   | Percentage of time that a signal is above a certain level. This model is a statistical model that describes the variation of the amplitude and phase of a signal due to multipath propagation.                 |
| Rician [34]                    | Fading                       | Indoor [35]    | Percentage of time that a signal is above a certain level. This model is similar to the Rayleigh fading model but assumes that there is a dominant line-of-sight path in addition to the multipath components. |
| WINNER II                      | Path loss, shadowing, fading | Indoor/Outdoor | Comprehensive model  |
| ITU-R P.1238-11                | Path loss, shadowing, fading | Indoor         | Comprehensive model  |
| ITU-R P.1411-11                | Path loss, shadowing, fading | Outdoor        | Comprehensive model  |
| Ray Tracing                    | Path loss, shadowing, fading | Indoor/Outdoor | Geometric model  |

## **B. Additional Considerations Applicable to FANETs**

Regarding large-scale propagation effects, FANETs have advantages over other types of mobile networks because of their ability to change altitude and move through a relatively less obstructed environment to achieve LoS or near LoS communications. However, FANET channel modeling presents additional conditions due to the mobility and variable altitude of the nodes involved. Some considerations should be made when modeling the FANET channel:

- **Doppler Shift:** Due to the high speeds at which the FANET nodes move, there will be a significant Doppler shift in the transmitted signals for air-to-air, air-to-ground, and air-to-satellite links.
- **Multipath:** It affects mostly the ground receivers, since the effects of multipath are largely determined by the local environment around the receiving antenna. This is because the signal may be reflected on various surfaces in the local geometry, causing it to arrive at the antenna through multiple paths [36]. However, FANET nodes may encounter multipath components due to reflections and scattering from the ground, buildings, and other obstacles when their height is below the height of surrounding structures.
- **Shadowing:** It is mostly generated by the environment surrounding the ground nodes. However, it has been shown that a fixed-wing aircraft body can self-induce shadowing on air-to-ground links while maneuvering [37].
- **Variations from LoS to Non-line-of-sight (NLoS) Propagation:** Due to the changes in position and altitude of FANET nodes, there may be a significant amount of time when the wireless links are NLOS.
- **Interference:** FANET nodes may also experience interference from other sources, such as other FANET nodes or other wireless systems operating in the same frequency band.

In general, the channel model for FANET should be designed to capture the dynamic and complex nature of the wireless links in such a network, and can be based on a combination of empirical measurements and theoretical modeling.

### **2.3.3 Low Power Wide Area Networks (LPWANs)**

LPWANs are wireless networks designed to provide long-range, low-power communication for IoT devices, such as sensors and smart meters. LPWANs use radio frequencies to transmit small amounts of data over long distances with low power consumption, and are ideal for applications that require long-range communication over a wide area, but do not require high bandwidth or low latency. Considering these characteristics, LPWANs and FANETs can complement each other in certain applications.

LPWANs and FANETs can be used together to provide enhanced functionality. For example, LPWANs can be used to transmit sensor data from ground-based IoT devices to UAVs in a FANET for further processing and analysis. The UAVs can then use their mobility and communication capabilities to transmit the processed data to a ground station or cloud-based server for storage and analysis [38]. Another application where LPWANs and FANETs can be combined is precision agriculture [39]. In this application, ground-based IoT sensors can provide data on soil moisture, temperature, and other environmental factors, while UAVs in a FANET can provide aerial images and real-time data on crop health and growth. The data from both the ground-based sensors and the UAVs can be processed and analyzed to provide insights and recommendations for farmers to optimize crop yields and reduce waste. Moreover, both networks can be used jointly to provide location services or low-data-rate real time communications (such as text messaging) in disaster scenarios [38,40].



There are several competing LPWAN technologies that use unlicensed spectrum such as Ingenu, Weightless (W, N and P), SigFox, or LoRaWAN [10,12]. Among these, LoRaWAN is one of the most adopted, because of its relative simplicity and low cost [10,41].

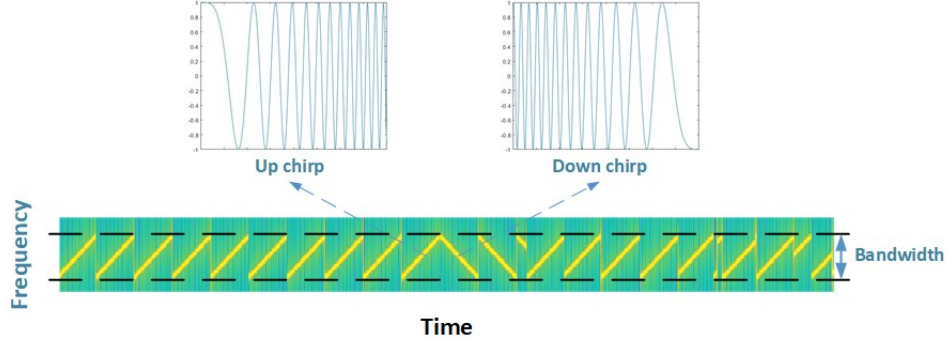
The Third Generation Partnership Project (3GPP) develops LPWAN standards that operate in licensed bands, such as EC-GSM-IoT, Narrow Band IoT (NB-IoT), enhanced Machine Type Communications (eMTC) [42], and Massive Machine-Type Communications (mMTC), which is the current IoT specification in 5G (Releases 16 and 17) [43]. These specifications have managed to reduce costs and energy consumption but have not been able to reach the adoption levels of other LPWAN technologies [10,13].

## **2.4 LoRa**

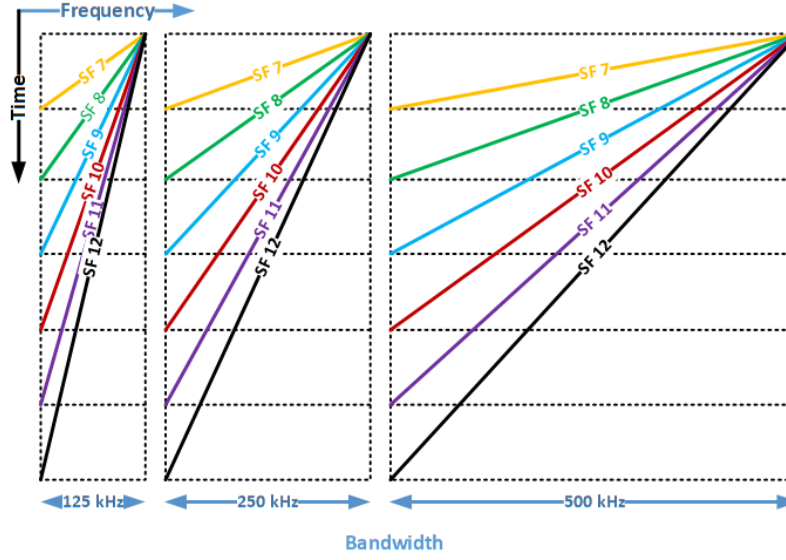
LoRa is an LPWAN technology that operates at the physical layer and provides long-range communication at low data rates. LoRa and LoRaWAN are often mentioned interchangeably; however, though complementary, they are two different things. LoRa is a PHY layer proprietary technology owned by Semtech [44], while LoRaWAN is an open network protocol specification—promoted by the LoRa Alliance—that uses LoRa as its physical layer but includes MAC and application layers [44,45]. LoRa uses a form of spread spectrum modulation called chirp spread spectrum (CSS) to achieve low power communications in the range of kilometers.

### **2.4.1 CSS Modulation**

Chirp spread spectrum (CSS) is a modulation technique that was first introduced in the 1940s to be used in military radar applications during World War II [46]. In CSS modulation, the data signal is first converted into symbols, where symbols are made of chirps. Chirps are sinusoidal signals whose frequency increases or decreases linearly within a certain range, defined by the bandwidth, and at a certain rate, defined by the spreading factor as illustrated in Figure 2.4.



(a) Up chirp and down chirp



(b) Sweep signal length

Figure 2.4: Chirp spread spectrum (CSS) modulation.

The modulation parameters are described next and summarized in Table 2.3.

## 2.4.2 Frequency

LoRa was conceived to transmit over unlicensed spectrum in industrial, scientific, and medical (ISM) bands. It currently operates in the 169 MHz, 433 MHz, 470 MHz, 490 MHz, 780 MHz, 868 MHz, 915 MHz and 2.4 GHz bands [47], subject to national and regional regulations.

## 2.4.3 Bandwidth (BW)

Bandwidth is the frequency range over which the chirps vary. It can take any of ten values ranging from 7.8 kHz to 1625 kHz, depending on the chipset and frequency band [48-52].

#### 2.4.4 Spreading Factor (SF)

The spreading factor represents the rate at which the frequency varies over the bandwidth. In other words, it defines the chirp (symbol) duration. The SF currently ranges from 5 to 12 [50-52] and the relationship between the SF value and the symbol duration is defined as follows:

$$T_s = \frac{2^{SF}}{BW[Hz]} [s], \quad (2.5)$$

where  $T_s$  is the symbol duration. Reciprocally, the symbol rate can be defined as:

$$R_s = \frac{BW[Hz]}{2^{SF}} \left[ \frac{\text{symbols}}{s} \right]. \quad (2.6)$$

According to the LoRa design, SF also represents the number of modulated bits per symbol, through which we can obtain the modulated bit rate:

$$R_m = SF \times R_s = SF \times \frac{BW[Hz]}{2^{SF}} \left[ \frac{\text{bits}}{s} \right]. \quad (2.7)$$

Considering that each symbol has the same duration, this tells us that the symbols are defined by the starting frequency of the chirp.

#### 2.4.5 Coding Rate (CR) Index

LoRa implements forward error correction (FEC) by adding redundancy bits to every 4 bits of data. The number of redundancy bits is given by the CR index, which can go from 1 to 4. Thus, [46] defines the *rate code* (generally known as coding rate) as:

$$\text{Rate Code} = \frac{4}{4 + CR}. \quad (2.8)$$

The coding rate and spreading factor are used to control the rate at which data is transmitted and the level of redundancy in the transmission. Higher coding rate index and spreading factor values yield a lower data rate, but a more robust transmission that is less susceptible to noise and

interference, thus resulting in a longer communication range. The data bit rate is the product of the modulated bit rate and the *rate code*, as follows:

$$R_b = R_m \times RateCode = SF \times \frac{BW[Hz]}{2^{SF}} \times \frac{4}{4 + CR} \left[ \frac{bits}{s} \right]. \quad (2.9)$$

#### 2.4.6 Transmission Power

The transmission power can reach up to 22 dBm, depending on the chipset selection and power amplifier configuration [48-52].

Two of the key parameters behind LoRa modulation are SF and BW. The relationship between these two factors defines the signal's data rate, range, and time on air. The higher the SF, the lower the transmission rate and the longer the range. Conversely, the lower the SF, the higher the transmission rate and the shorter the range.

Table 2.3: Summary of LoRa modulation parameters.

| Parameter      | Magnitude/Range | Chip   | Reference  |
|----------------|-----------------|--|------------|
| Frequency      | 137 – 175 MHz   | SX1276/77/78/79                              | [48]       |
|                | 410 – 525 MHz   | SX1276/77/78/79                              | [48]       |
|                | 862 – 1020 MHz  | SX1276/77/79                                 | [48]       |
|                | 860 – 1020 MHz  | SX1272/73                                    | [49]       |
|                | 410 – 810 MHz   | SX1268                                       | [50]       |
|                | 150 – 960 MHz   | SX1261/2                                     | [51]       |
|                | 2.4 GHz         | SX1280/SX1281                                | [52]       |
| Bandwidth (BW) | 7.8 kHz         | SX1276/77/78/79, SX1268, SX1261/2            | [48,50,51] |
|                | 10.4 kHz        | SX1276/77/78/79, SX1268, SX1261/2            | [48,50,51] |
|                | 15.6 kHz        | SX1276/77/78/79, SX1268, SX1261/2            | [48,50,51] |
|                | 20.8 kHz        | SX1276/77/78/79, SX1268, SX1261/2            | [48,50,51] |
|                | 31.2 kHz        | SX1276/77/78/79, SX1268, SX1261/2            | [48,50,51] |
|                | 41.7 kHz        | SX1276/77/78/79, SX1268, SX1261/2            | [48,50,51] |
|                | 62.5 kHz        | SX1276/77/78/79, SX1268, SX1261/2            | [48,50,51] |
|                | 125 kHz         | SX1276/77/78/79, SX1272/73, SX1268, SX1261/2 | [48-51]    |
|                | 250 kHz         | SX1276/77/78/79, SX1272/73, SX1268, SX1261/2 | [48-51]    |
|                | 500 kHz         | SX1276/77/78/79, SX1272/73, SX1268, SX1261/2 | [48-51]    |
|                | 203 kHz         | SX1280/SX1281                                | [52]       |

| Parameter             | Magnitude/Range | Chip  | Reference |
|-----------------------|-----------------|---|-----------|
|                       | 406 kHz         | SX1280/SX1281   | [52]      |
|                       | 812 kHz         | SX1280/SX1281   | [52]      |
|                       | 1625 kHz        | SX1280/SX1281   | [52]      |
| Spreading Factor (SF) | 5               | SX1268, SX1261/2, SX1280/SX1281                             | [50-52]   |
|                       | 6 - 9           | SX1276/77/78/79, SX1272/73, SX1268, SX1261/2, SX1280/SX1281 | [48-52]   |
|                       | 10 - 12         | SX1276/78/79, SX1272, SX1268, SX1261/2, SX1280/SX1281       | [48-52]   |
| Coding Rate (CR)      | 1 (4/5)         | SX1276/77/78/79, SX1272/73, SX1268, SX1261/2, SX1280/SX1281 | [48-52]   |
|                       | 2 (4/6)         | SX1276/77/78/79, SX1272/73, SX1268, SX1261/2, SX1280/SX1281 | [48-52]   |
|                       | 3 (4/7)         | SX1276/77/78/79, SX1272/73, SX1268, SX1261/2, SX1280/SX1281 | [48-52]   |
|                       | 4 (4/8)         | SX1276/77/78/79, SX1272/73, SX1268, SX1261/2, SX1280/SX1281 | [48-52]   |
| Transmission Power    | -4 to 20 dBm    | SX1276/77/78/79   | [48]      |
|                       | -1 to 20 dBm    | SX1272/73   | [49]      |
|                       | -17 to 22 dBm   | SX1268  | [50]      |
|                       | -17 to 22 dBm   | SX1261/2  | [51]      |
|                       | -18 to 12.5 dBm | SX1280/SX1281   | [52]      |

## 2.4.7 LoRa Frame Format

The LoRa specification also defines frame formats. As with any protocol data unit, frame formats are intended to allow communication between peer elements in a layered network, for which they separate the protocol control information (overhead) from the payload, in a standardized way. A LoRa frame consists of a preamble, a sync word, a header, the payload, and a cyclic redundancy check (CRC) [48-52], as shown in Figure 2.5.

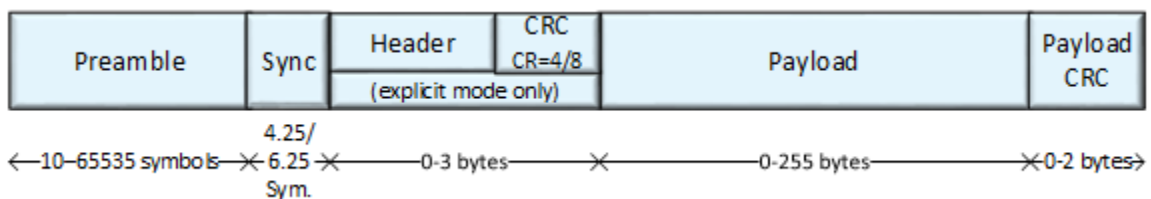


Figure 2.5: LoRa frame format

## 2.5 IEEE 802.11

IEEE 802.11 is a set of standards that define the technology behind wireless local area networks (WLANs) [53]. These standards are commonly referred to as Wi-Fi and are used by devices to communicate with each other over a wireless network. The IEEE 802.11 standards specify different data rates, frequency bands, and modulation schemes that can be used to transmit data wirelessly between devices. Some of the most commonly used standards include 802.11a, 802.11b, 802.11g, 802.11n, 802.11ac, and 802.11ax. A summary of IEEE 802.11 standards and their characteristics is presented in Table 2.4.

Table 2.4: Summary of IEEE 802.11 standards and their characteristics.

| Standard | Frequency Band | Maximum Data Rate | Modulation Techniques | Medium Access Control | Routing Protocols |
|----------|----------------|-------------------|-----------------------|-----------------------|-------------------|
| 802.11a  | 5 GHz          | 54 Mbps           | OFDM                  | CSMA/CA               | None              |
| 802.11b  | 2.4 GHz        | 11 Mbps           | DSSS                  | CSMA/CA               | None              |
| 802.11g  | 2.4 GHz        | 54 Mbps           | OFDM, DSSS            | CSMA/CA               | None              |
| 802.11n  | 2.4/5 GHz      | 600 Mbps          | MIMO-OFDM             | CSMA/CA               | None              |
| 802.11ac | 5 GHz          | 6.9 Gbps          | MIMO-OFDM             | CSMA/CA               | None              |
| 802.11ax | 2.4/5/6 GHz    | 9.6 Gbps          | MIMO-OFDM, MU-MIMO    | CSMA/CA               | None              |
| 802.11p  | 5.9 GHz        | 27 Mbps           | OFDM                  | CSMA/CA               | None              |
| 802.11s  | 2.4/5 GHz      | Variable          | Variable              | CSMA/CA               | HWMP              |

Note that the actual data rates that can be achieved on a wireless network will depend on a variety of factors, including the distance between devices, the presence of obstacles or interference, and the capabilities of the devices themselves.

In addition to specifying data rates and frequency bands, the IEEE 802.11 standards also include provisions for security and quality of service (QoS). Security is provided through the use of encryption algorithms such as WEP, WPA, and WPA2, while QoS is used to prioritize different types of traffic on the network.

### **2.5.1 IEEE 802.11p**

IEEE 802.11p is a standard for wireless communication in vehicular environments, designed to support applications such as collision avoidance, traffic management, and infotainment systems in vehicles [54]. It operates in the 5.9 GHz frequency band, which has been set aside for this purpose by regulatory bodies in many countries. The maximum data rate for IEEE 802.11p is 27 Mbps, which is lower than some of the other IEEE 802.11 standards, but is still sufficient for many vehicular applications. The standard uses OFDM modulation, which allows for efficient use of the available bandwidth, and the CSMA/CA medium access control protocol, which is used in all the IEEE 802.11 standards.

### **2.5.2 IEEE 802.11s**

IEEE 802.11s is a standard for wireless mesh networks, which allows multiple wireless access points to work together to create a single, seamless network. Unlike traditional Wi-Fi networks, where devices communicate directly with a single access point, wireless mesh networks allow devices to communicate with multiple access points in order to find the best path for data transmission.

The 802.11s standard specifies the use of the Hybrid Wireless Mesh Protocol (HWMP) for managing the flow of data between different nodes on the mesh network. HWMP is inspired by AODV and performs routing at the link layer based on MAC addresses [55]. The maximum data rate for wireless mesh networks can vary depending on the number of nodes in the network and the quality of the wireless links between them.

### **2.5.3 IEEE 802.11ax**

IEEE 802.11ax, also known as Wi-Fi 6, is the latest wireless networking standard that builds upon previous IEEE 802.11 standards, such as 802.11ac (Wi-Fi 5) and 802.11n. The goal of IEEE

802.11ax is to improve the speed, capacity, and overall performance of Wi-Fi networks, especially in dense environments with many devices competing for network resources [56].

Some key features and benefits of IEEE 802.11ax include:

1. Increased data rates: IEEE 802.11ax supports maximum data rates of up to 9.6 Gbps.
2. Improved spectral efficiency: IEEE 802.11ax introduces a new modulation scheme called 1024-QAM, which enables more efficient use of the wireless spectrum. It also supports narrower channel widths of 20 MHz, 40 MHz, and 80 MHz, which can help reduce interference between Wi-Fi networks.
3. MU-MIMO: IEEE 802.11ax introduces Multi-User Multiple Input Multiple Output (MU-MIMO) technology, which allows multiple devices to simultaneously communicate with the access point using multiple antennas. This increases network capacity and reduces latency, especially in environments with many devices.
4. OFDMA: IEEE 802.11ax introduces Orthogonal Frequency Division Multiple Access (OFDMA), which divides a single Wi-Fi channel into multiple sub-channels to enable multiple users to transmit data simultaneously. This can help improve network efficiency and reduce latency.
5. Target Wake Time (TWT): IEEE 802.11ax includes a new power-saving feature called TWT, which allows devices to schedule their wake-up times and communicate with the access point only when necessary. This can help extend battery life in devices such as smartphones and IoT devices.

## **2.6 FANET Mobility**

FANET mobility is made possible through the use of advanced control systems and navigation technologies that allow UAVs to fly autonomously and maintain formation while communicating



with other nodes in the network. One of FANETs' challenges is the high mobility of UAVs, which makes it difficult to maintain stable communication links between them. However, this ability to move with a relatively higher level of freedom than other forms of MANET can be seen as an opportunity instead of a problem, where nodes can move and reposition themselves to reconfigure a network or to recover connectivity. In this section, the FANET mobility issue is broken down into mobility objectives, mobility models, and the optimization approach to the mobility problem.

### **2.6.1 Mobility objectives**

Mobility objectives can be described as optimization problems that are dependent on various factors such as: application, environmental conditions, and available resources. Optimization techniques can be useful in achieving the following objectives:

1. Optimal positioning: Where to go and why.
2. Optimal trajectory determination: How to get there and why.
3. Optimal agent selection: Which UAVs should get there and why.

Regardless of the specific application, a common requirement for FANETs is to maximize network uptime, in the literal and figurative senses of the expression.

### **2.6.2 Mobility models**

A mobility model is a mathematical model that describes the movement patterns of the nodes in a network over time. Mobility models can be used to model movement for FANET simulation and to achieve mobility optimization goals in practical implementations. In most of the research, mobility models are used to simulate UAV movement in order to assess communication protocols performance, but less have been used as a way to achieve coordinated motion in order to accomplish a common objective.

Selecting a specific model depends, among other factors, on the application, on the required level of detail, and on the type of UAV to be used, considering that not every type of UAV can perform the same kind of movements. A summary of different mobility models presented in [25-27] is provided next:

- **Random Walk Model:** The Random Walk model is one of the simplest mobility models used for FANETs. In this model, the UAVs move in any direction with a constant velocity. The direction of the UAVs' movement is determined randomly at each step. This model assumes that the UAVs move independently of each other and that there are no external factors affecting their movement.
- **Random Waypoint Model (RWP):** The Random Waypoint model is another widely used mobility model for FANETs. In this model, the UAVs move in straight lines from one point to another, with varying velocities. The direction and speed of the UAVs' movement are chosen randomly. When the UAVs reach their destination, they pause for a random amount of time before moving on to the next destination. This model is useful for simulating mission-based scenarios, where the UAVs must reach specific waypoints.
- **Gauss-Markov (GM):** In this stochastic model, each UAV has a speed and a direction. The speed and direction are updated at each time step, based on a random process. The direction is chosen randomly from a uniform distribution over the range of 0 to  $2\pi$  radians. The speed of the UAVs follows a first-order autoregressive process, meaning that it depends on its previous speed and a random error.
- **Mission Plan (MP):** This deterministic model is based on pre-defined flight plans or routes that are created before the UAVs are deployed. The Mission Plan mobility model is often used in military and surveillance applications, where the UAVs are required to fly specific

routes to collect data or perform other tasks. One limitation of the Mission Plan mobility model is that it does not capture the unpredictability and randomness of real-world mobility patterns.

- **Semi-Random Circular Movement (SRCM):** In this model, the UAVs move in circular patterns around a fixed center point. In the SRCM mobility model, the UAVs move along arch trajectories with a radius that varies randomly.
- **Paparazzi Mobility Model (PPRZM):** This stochastic model is named after the Paparazzi UAV project, which is an open-source autopilot system used in FANETs [57]. PPRZM is based on a state machine with five possible states or movement patterns: Eight-figure, Stay-at, Waypoint, Oval, and Scan, where each type of movement has a different probability of occurrence.
- **Reference Point Group Mobility (RPGM) model:** In this stochastic group mobility model, UAVs form clusters. Each cluster has a central reference point that can be logical, or a UAV selected as cluster head. The reference point moves using an RWP model, while the other UAVs within the cluster move around the center.

## **2.7 Optimization Approach to the Mobility Problem**

Optimization is the process of improving a system, process, or design in order to maximize its efficiency. It is an essential concept in various fields, including engineering, economics, finance, computer science, and management. The goal of optimization is to find the best possible solution to a problem while satisfying certain constraints. The first step in this process is to identify the objective and a way to measure it quantitatively. The next step is to identify the variables on which the objective depends, also called unknowns. As the objective is a function of the variables, it is also called objective function. In some cases, the variables are subject to constraints, and

identifying those constraints is the third step. Constraints are the limitations or restrictions that must be considered when developing a solution. These constraints place limitations on the values that the variables can take, and they are typically specified by equality or inequality relationships. These first three steps are known as model development or modelling [58].

Once the model has been developed, the next step is to find a solution. The solution is the value or set of values that the variables can take that maximize or minimize the objective function. This involves using various optimization techniques to find the best possible solution that satisfies the objectives and constraints. Optimization techniques could include linear programming, nonlinear programming, dynamic programming, and heuristic methods.

The optimization approach can be applied to FANET mobility by defining an objective that is a function of the UAVs' positions. Ideally, the solutions will be the positions that minimize or maximize the objective function. Furthermore, the mechanism employed to find the solution or solutions can provide locations to dynamically position the UAVs. As in many other fields, some FANET problems may present multiple conflicting objectives. In such cases, the optimal solution must achieve a suitable trade-off between these objectives.

### **2.7.1 Global and Local Optimization**

Global optimization refers to finding the best possible solution within the entire search space. This search space can be very large, and finding the optimal solution can be very challenging. Global optimization algorithms try to explore a large part of the search space in order to identify the global optimum, using techniques such as random search, genetic algorithms, or simulated annealing. However, these methods may require significant computational resources, and they are not always guaranteed to find the global optimum.

In some cases, it is good enough to find only a local solution, which is a solution that is optimal within a limited region of the search space or in the vicinity of a current solution. Local solutions are typically not globally optimal, meaning that they are not necessarily the best possible solution for the entire search space. Instead, they are the best possible solution within the specific neighborhood that the local optimization algorithm is exploring.

### **2.7.2 Single-Objective and Multiobjective Optimization**

Single-objective optimization is an optimization problem that involves finding the best solution to a problem by optimizing a single objective function. In other words, it seeks to minimize or maximize a single objective.

Alternatively, multiobjective optimization is an optimization problem that involves finding the best solution to a problem by optimizing multiple objectives simultaneously. In this case, the objective functions are usually conflicting and cannot be optimized independently at the same time, so the goal is to find a set of solutions that achieve a reasonable trade-off between the objectives. Multiobjective optimization problems have also multiple solutions [59].

### **2.7.3 Single-Solution and Multiple-Solution Optimization**

Single solution optimization is an optimization problem where only one solution is required for a single objective function.

In contrast, multiple-solution optimization is an optimization problem where multiple solutions may satisfy the objective function or multiple objective functions simultaneously. The goal is to find a set of optimal solutions that represent a trade-off between the different objectives. Multiple solution-optimization is commonly used in problems where the objective function has multiple local minima or maxima, or where multiple objectives need to be satisfied.

### 2.7.4 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is an optimization algorithm that was inspired by the social behavior behind the coordinated flight of a flock of birds [60,61]. It works by simulating a swarm of particles that move in a search space to find the optimal solution. Each particle in the swarm represents a potential solution, and its position in the search space is adjusted according to its own previous best solution and the best solution found by all the particles in the swarm. The equations that govern PSO are:

$$v_i(t + \Delta t) = wv_i(t) + c_1r_1(p_i - x_i(t)) + c_2r_2(g - x_i(t)), \quad (2.10)$$

where  $v_i(t)$  is the velocity of the  $i$ -th particle at time  $t$ ,  $w$  is the inertia weight,  $p_i$  is the personal best position ( $pbest$ ) of the  $i$ -th particle,  $x_i(t)$  is the current position of the  $i$ -th particle at time  $t$ ,  $g$  is the global best position ( $gbest$ ),  $c_1$  and  $c_2$  are the cognitive and social learning parameters, and  $r_1$  and  $r_2$  are random numbers between 0 and 1.

$$x_i(t + \Delta t) = x_i(t) + \Delta t \cdot v_i(t + \Delta t), \quad (2.11)$$

where  $x_i(t + \Delta t)$  is the updated position of the  $i$ -th particle at time  $t + \Delta t$ .

The algorithm starts with randomly initializing the positions and velocities of the particles in the search space. Then, for each iteration, the velocity and position of each particle are updated based on the equations above. The personal best position of each particle is updated if a better solution is found, and the global best position is updated if a particle finds a better solution than the current global best. The algorithm continues until a stopping criterion is met, such as a maximum number of iterations or a desired level of fitness.

## **2.8 Related work**

There has been a significant amount of research in FANETs in recent years, addressing various challenges and issues related to their operation. Researchers have proposed various protocols and algorithms for efficient routing, data dissemination, and coordination among UAVs. These protocols and algorithms are intended to optimize the performance of the network while minimizing the energy consumption and maximizing the network lifetime.

### **2.8.1 FANET Architectures Involving LoRa or LoRaWAN**

Regarding the use of LoRa or LoRaWAN, most works refer to UAV-aided wireless sensor networks (WSNs) with single or independent UAVs [62-69]. WSNs are sets of sensors that communicate with each other by forming an ad hoc network. By this definition, WSNs can be considered a subset of IoT. However, even though WSNs and FANETs have a mesh nature in common, a WSN aided by a single UAV is not considered a FANET by the definition presented at the beginning of this chapter.

### **2.8.2 FANET Mobility**

Reference Point Group Mobility (RPGM) is used as the mobility model for performance evaluations in [70], which are carried out through simulations in Network Simulator (NS)-3. These simulations show that the inclusion of an RSSI-based factor results in an improvement in network throughput and a decrease in the number of route changes, with little to no impact on end-to-end delay, especially for a mobile scenario.

An algorithm to find the optimal position of a UAV-mounted relay node is presented in [71] to maximize the throughput between any pair of fixed ground nodes within a wireless mesh network. To achieve this, nodes —whose positions are known— are first clustered in a way so that nodes that connect within a certain RSSI margin are part of the same cluster. Then, the placement

optimization is performed by separating the positioning problem into horizontal and vertical placement and iterating between the two. For the horizontal placement, clusters are considered single entities around their center of gravity, and the UAV is positioned to maximize throughput—as a function of signal-to-noise ratio—for the worst wireless link. Subsequently, vertical positioning is determined to maximize throughput in the same manner. As this work focuses on static mesh networks, routing is done by means of an optimized link state routing protocol (OLSR). Simulations for up to 20 ground nodes are implemented using a 3D map. Also, an experimental performance assessment is presented with four ground nodes separated into two clusters of two nodes each.

Concerning mobility models in FANETs that use LoRa, a Connection Recovery and Maintenance algorithm (CRM) is proposed in [38,72], where a two-dimensional mobility problem is divided into four mobility modes. A Virtual Spring Force (VSF) mobility algorithm is used to handle proximity between nodes to avoid collisions and to maintain distance within communications range. Correspondingly, if the forces are in balance, the UAV goes into stationary mode. Then, if a path to the base station is lost, the UAV goes into network recovery mode, where it moves in the direction of the base station; and, if the connection is lost with a ground node, movement prediction is used alongside the spring force mobility to try to reestablish it, based on the ground node's last known position, direction, and speed.

A two-phase particle swarm mobility model (PSMM) is proposed in [73] to generate fixed waypoint trajectories that allow UAVs to maintain a stable formation in a swarm during a certain simulation time. In the first phase, a variation of PSO is employed to determine each UAV's next waypoint. The proposed PSO algorithm excludes the personal component and uses the swarm's center of gravity at each time step as the global best position for the social component. In the



second phase, the previously generated waypoints are evaluated for excessive closeness and adjusted to avoid collisions, while maintaining formation. The algorithm performance is compared with those of RPGM, Manhattan, and RWP in terms of spatial correlation, temporal correlation, and path availability.

In [74], the authors propose a PSO-based algorithm for dynamic positioning of UAVs in wireless sensor networks (WSNs). The proposed algorithm is intended to optimize the placement of UAVs to collect information from a set of ground sensors located at previously known positions and transmit the collected information to a ground base station. The problem is considered as a constrained, multiobjective optimization, where the algorithm dynamically adjusts the position of the UAVs based on three factors: the value of their collected sensory information, the quality of the sensor-ground base station communications path, and the existence of a path from each UAV to the ground base station. The objective function, which is evaluated at the ground base station, is defined as the product of the three factors for each UAV, and aggregated for all UAVs. The PSO algorithm is also executed at the ground base station in a centralized manner. As opposed to the study in [73], this work does not consider each particle to be a UAV, but a set of candidate positions for each UAV. Therefore, the number of particles is not necessarily equal to the number of UAVs and is, in fact, larger. It can be observed that an entire PSO simulation is executed at the base station before actually transmitting new positions to the UAVs. Additionally, it is assumed that every UAV will have a path to the control station at each new optimal position. Given the centralized nature of the proposed algorithm, in the case of a real implementation, if such path does not exist, the UAV would become isolated and potentially lost. Finally, a link state routing algorithm is applied to establish a path from a set of ground sensors to a ground base station, through the UAVs in the FANET.

## **Chapter 3**

# **Simulation Framework Development**

One of FANETs applications is to provide communication coverage over a wide area. These networks can be used to provide seamless coverage in scenarios where ground-based communication infrastructure is unavailable or unreliable, such as in disaster response, military operations, or remote areas. Seamless coverage refers to the provision of uninterrupted communication coverage even when devices move in and out of each other's range [29,38,39,72,75,76]. This is achieved through the use of multi-hop communication, where UAVs act as relays to forward data packets between nodes that are out of range of each other. As a UAV moves out of range of one node, it automatically connects to the nearest available node and continues to provide communication coverage. This ensures that the network maintains its coverage even as UAVs move around or are added or removed from the network.

This chapter focuses on the development of a simulation framework for a scenario where seamless coverage of ground sensor nodes is achieved through mobility optimization. The model has been implemented in MATLAB R2022a, using the UAV and WLAN toolboxes taking communications and mobility into consideration. The MATLAB main script is included in APPENDIX A. The remainder of the chapter is organized as follows: The architecture of the proposed network is discussed in Section 3.1. Section 3.2 deals with the considerations and assumptions that have been made for the simulation. Section 3.3 provides the definition of the objective function followed by Section 3.4, in which the mobility and the implementation of the PSO algorithm are discussed. Finally, Section 3.5 presents additional details on the model configuration and a summary of the simulation parameters.

### 3.1 System Architecture

From the communications perspective, the system follows a single-cluster mesh topology as described in Figure 2.1. The network is composed of two layers: (i) the ground layer consisting of ground nodes and a control station, and (ii) the air layer consisting of a swarm of UAVs, specifically quadrotor drones., as illustrated in Figure 3.1.

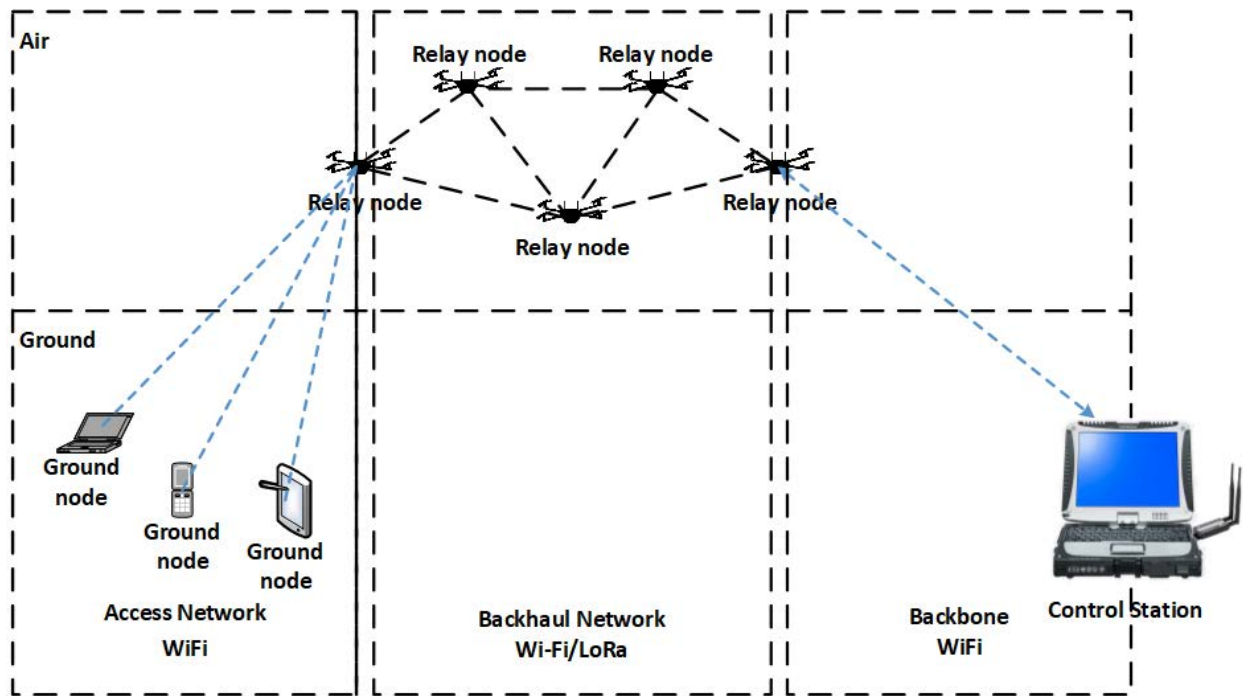


Figure 3.1: System architecture.

The UAVs communicate with the ground nodes using Wi-Fi, as it is a more ubiquitous technology among end users. On the other hand, the UAVs communicate with each other using LoRa or Wi-Fi technologies. The network elements are described next:

- Ground nodes: User equipment with Wi-Fi transceivers.
- UAVs: Quadrotor drones equipped with Wi-Fi and LoRa transceivers.
- Control station: In the simulation environment, it is intended to receive information from the ground nodes that has been relayed through the UAV FANET.

## 3.2 Considerations and Assumptions

The following general assumptions are made while the simulation framework:

- The ground nodes are at static positions inside a square grid during the simulation.
- The UAV flight dynamic model is limited to changes in position and velocity.
- The ground nodes'  $x$  and  $y$  coordinates are independent random variables uniformly distributed along their respective axes.
- The control station is at a fixed position.
- A flat earth model is implemented, considering that the maximum separation between UAVs and ground nodes will not exceed a few tens of kilometers [34].
- The simulation time step ( $\Delta_t$ ) is set to 1 s.
- For radio propagation analysis, the UAVs are assumed to be stationary at each time step.
- Wi-Fi nodes and access points (APs) are previously associated, meaning that the nodes have established a connection to the APs in the past.
- UAVs within range of each other can share their positions and their received power levels.
- PSO random factors  $r_1$  and  $r_2$  are uniformly distributed numbers between 0 and 1.
- Communication is engaged once the UAVs have attained stable positions.

Further considerations regarding communications are described in the following subsections.

### 3.2.1 Communications

As the objective function is built on received power and other communications parameters, this section elaborates on concepts that will be used for further improvement of the objective function.

### A. Physical Layer

The physical layer relies on IEEE 802.11ax, also known as Wi-Fi 6, and LoRa protocols. Both protocols will be considered as defined by their standardization body and provider, respectively. Therefore, this section will focus on the propagation model that will be used to calculate the received signal levels from the ground nodes and from other UAVs at each UAV. Moreover, the ground-to-air and air-to-air received signal levels will be used subsequently to define and evaluate the objective function for the optimization algorithm.

The path loss models include many empirical parameters. However, they all share the form of a log-distance path model with shadowing, which is a form of log-normal propagation. Therefore, for the purpose of the proposed model, the following large-scale propagation models are implemented: free space path loss, ray tracing, and log-normal path loss with parameters for a rural environment. In all cases, the path loss can be described using the general form of the log-normal shadowing model shown in Equation (2.4). The path loss is used to calculate the received power level at each UAV in dBm according to Equation (3.1):

$$R(x_i)_j[dBm] = PT_j[dBm] + G_t[dB] + G_r[dB] - PL(d_{A2G})[dB] \quad (3.1)$$

where  $R(x_i)_j[dBm]$  is the signal power received by UAV  $i$ , at time  $t$ , and position  $x_i$ , from the  $j$ -th ground node;  $PT_j$  is the transmission power from ground node  $j$ ;  $G_t$  and  $G_r$  are the transmission and reception antenna gains, respectively; and  $PL(d_{A2G})$  is the path loss over the distance between UAV  $i$  and ground node  $j$ .

### B. Routing

Once a ground node has been found, a communication path to the control station must be established. For this, a mechanism to find the shortest path between the UAV covering the source

ground node and the control station is needed. Since all received power levels in mW are positive, Dijkstra's algorithm can be implemented [77]. Here, the UAVs are considered vertices and the connections between them are edges in a graph. The graph is characterized by an adjacency matrix that represents the costs between nodes. Initially, a matrix is generated to record the received power levels at each node, accounting for interactions with all other nodes. An edge is established between two vertices if the received power level exceeds the defined receiving threshold. When there is no connecting edge between two vertices, the cell value is set to infinity,  $\infty$ . In the next step, the adjacency matrix is constructed, treating all edges linking two vertices as having a uniform cost of one. As a result, the graph becomes unweighted. Furthermore, as the transmission and reception parameters are the same for all nodes, this approach makes the graph undirected. However, if different propagation parameters were to be used for each node, the procedure is the same as before but would result in a directed graph.

The implementation of Dijkstra's algorithm in MATLAB code is available on [78] under a General Public License (GNU) v3.0 and presented in APPENDIX B. The function *Dijkstras* receives the adjacency matrix,  $G$ , the source node,  $S$ , and the target node,  $T$ , as inputs, while it returns a shortest path vector and its cost as outputs. The shortest path vector is a sequence of the traversed nodes, starting at the source node and ending at the target node. If there is no valid path from the source node to the target node, the returned path is an empty vector with infinite cost.

For the purposes of this model, the nodes have full knowledge of the network, making the described approach a case of link state routing.

### 3.3 Definition of the Objective Function

Initially, the objective function is formulated as the sum of the received power levels from the ground nodes that exceed a certain threshold, in this case, the sensitivity of the UAV's Wi-Fi

receiver. Additionally, if there are no signals whose power is above the threshold, the function takes the maximum value among all the signals from the ground nodes. The objective function of UAV  $i$ , at position  $x_i$ ,  $f(x_i)$ , is evaluated at each UAV, at every simulation step, according to Equation (3.2):

$$f(x_i) = \begin{cases} \sum_{j=1}^n \begin{cases} R(x_i)_j, & R(x_i)_j \geq \tau \\ 0, & R(x_i)_j < \tau \end{cases}, & |A| > 0 \\ \max_{1 \leq j \leq n} R(x_i)_j, & |A| = 0 \end{cases} \quad (3.2)$$

where  $n$  is the number of ground nodes,  $R(x_i)_j$  is the received power level from the  $j$ -th ground node, and  $A$  is the set of  $R(x_i)$  values above the threshold  $\tau$ :

$$A = \{R(x_i) | R(x_i) \geq \tau\}. \quad (3.3)$$

Note that  $|A|$  represents the number of elements—or cardinality—of set  $A$  in Equation (3.2).

As it can be inferred from Equations (2.1)-(2.3), the received power levels vary exponentially with the distance between transmitter and receiver. Therefore, there are situations where adding the received power levels from multiple ground nodes above the threshold does not necessarily improve the objective function value, when compared to a position closer to a single node. This is because the received power level from a single node at a closer distance can be significantly higher than the sum of the received power levels from multiple nodes, even if they are individually above the threshold. To improve the value of the objective function when more than one node is covered, an exponential reward factor was introduced as described in Equation (3.4).

$$f(x_i) = \begin{cases} 2^{|A|-1} \sum_{j=1}^n \begin{cases} R(x_i)_j, & R(x_i)_j \geq \tau \\ 0, & R(x_i)_j < \tau \end{cases}, & |A| > 0 \\ \max_{1 \leq j \leq n} R(x_i)_j, & |A| = 0 \end{cases} \quad (3.4)$$

where  $A$  remains as defined in Equation (3.3).

The base-2 exponential factor achieves an improvement of the objective function. However, it does not adapt well to large changes in grid size, as the aggregated received power levels at each UAV are dependent on grid size and ground node density. If the gain value is too low, the UAVs might return to the previous best position with a single covered ground node, even if they find more than one ground node at a different position. Conversely, if the factor is too high, the UAV might find many ground nodes but become isolated from the swarm and the control station. Therefore, it is essential to improve the value of the objective function when more than one ground node is located so that it is comparable to or better than the one obtained when a single ground node is found. It can be observed from Equation (2.1) that the improvement factor should be proportional to the distance that separates two points on the grid: the point where the UAV locates more than one ground node and a position where it finds only one node from which it receives a signal with a substantially higher power. As the ground node positions are unknown for the UAVs, the average distance between any two points on a square grid has been derived according to the approach depicted in Figure 3.2.



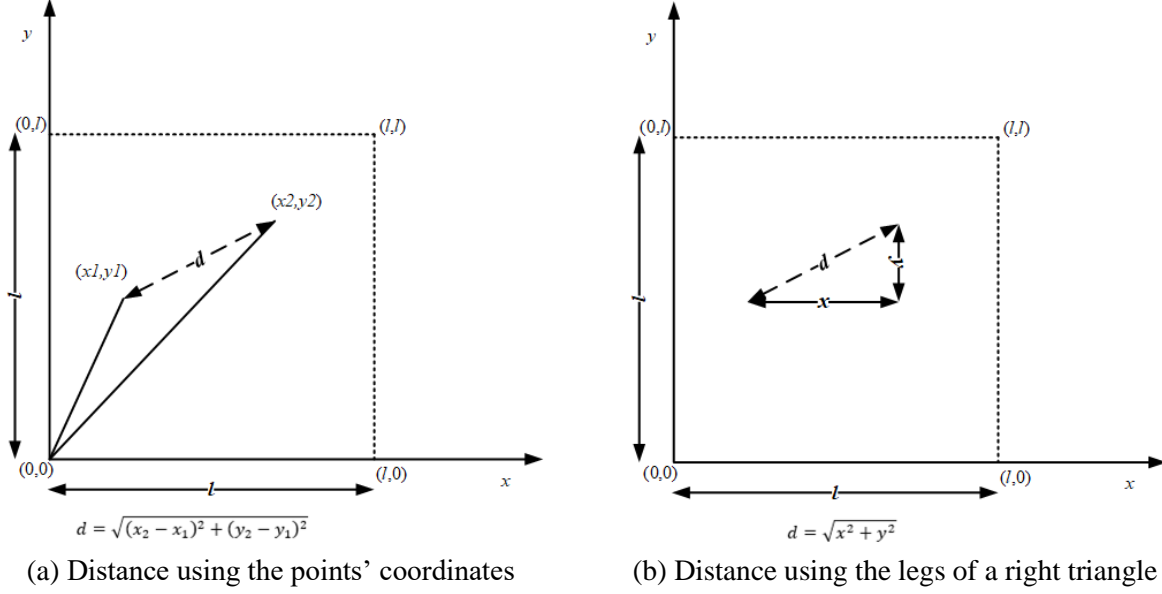


Figure 3.2: Approaches to distance calculation between two points in a square grid of side  $l$ .

The approach illustrated in Figure 3.2(a) involves solving four iterated integrals of the distance as a function of four uniform random variables, namely,  $x_1$ ,  $x_2$ ,  $y_1$ , and  $y_2$ , which is a long and demanding task. Using the approach shown in Figure 3.2(b), the required distance  $d$  is equal to  $\sqrt{x^2 + y^2}$ . This value must be weighted by the average remaining length in each axis, which is  $(l - x)$  horizontally and  $(l - y)$  vertically. Consequently, the average distance between any two points becomes the quotient of two iterated integrals of functions of two variables:

$$\bar{d} = \frac{\int_0^l \int_0^l (l - x)(l - y)(\sqrt{x^2 + y^2}) dx dy}{\int_0^l \int_0^l (l - x)(l - y) dx dy} \quad (3.5)$$

$$\bar{d} = \frac{\frac{2 + \sqrt{2} + 5 \ln(1 + \sqrt{2})}{60} l^5}{\frac{l^4}{4}}$$

$$\bar{d} = \frac{2 + \sqrt{2} + 5 \ln(1 + \sqrt{2})}{15} l \cong 0.5214l$$

The integration of the numerator was carried out for indefinite integrals using the online tool Wolfram Alpha [79], but the resulting functions were manually evaluated by the author at each

iteration. The result shows that the expected distance between any two uniformly distributed points on a square grid is dependent on the grid size only and not on the number of ground nodes, which is convenient for the proposed scenario, as the grid size is known for the UAVs, but the number of ground nodes is not. However, the effective distance is that between the UAV and a ground node. Thus, the UAV height must be accounted for, as shown in Figure 3.3.

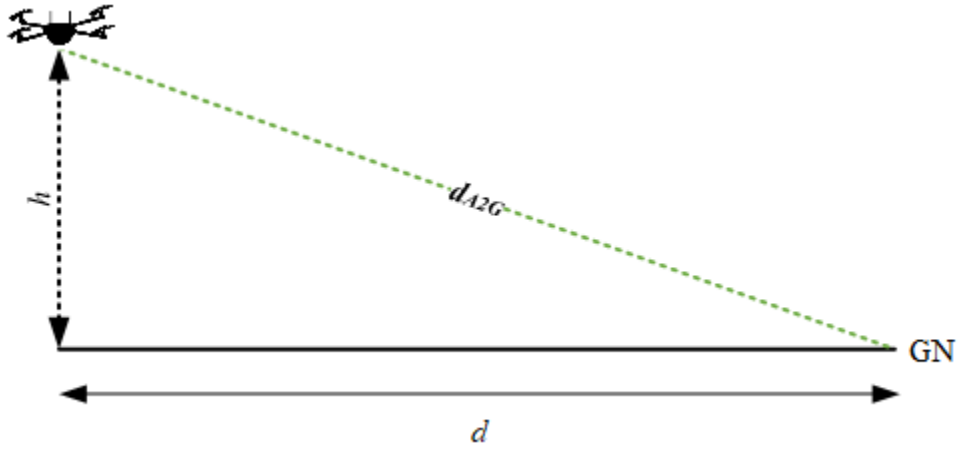


Figure 3.3: Air-to-ground distance at height  $h$  for the expected ground-to-ground distance  $d$ .

As power decays by a quadratic exponent in free space, the rewarding factor for finding more than one ground node must play an inverse role, on account of which, the updated reward factor plays the role of a gain. Thus, Equation (3.5) is modified as Equation (3.6) given below:

$$f(x_i) = \begin{cases} G_{mn} \sum_{j=1}^n \begin{cases} R(x_i)_j, & R(x_i)_j \geq \tau \\ 0, & R(x_i)_j < \tau \end{cases}, & |A| > 1 \\ \sum_{j=1}^n \begin{cases} R(x_i)_j, & R(x_i)_j \geq \tau \\ 0, & R(x_i)_j < \tau \end{cases}, & |A| = 1 \\ \max_{1 \leq j \leq n} R(x_i)_j, & |A| = 0 \end{cases} \quad (3.6)$$

$$G_{mn} = (0.5214l)^2 + h^2$$

where  $G_{mn}$  is the gain applied to the objective function for finding multiple nodes, and  $A$  remains as defined in Equation (3.3).

This approach turns out to be convenient, as it does not depend on ground node density, which is an unknown value for the UAVs, but only depends on the grid size and UAV height, which are indeed known values for the drone at any moment.

### **3.3.1 Multiobjective Optimization**

The main goal of this research work is to maximize the ground node coverage, as well as to maintain end-to-end communication with the control station through the UAV mesh network. Increasing coverage without maintaining communication to the distant control station would oppose the whole purpose of a FANET. Therefore, it is crucial to balance the competing objectives and handle the research problem using multiobjective optimization. This has been achieved by using a method that rewards or penalizes the objective function based on its behavior relative to the multiple problem objectives.

The gain applied to the objective function in Equation (3.6) is large because it accounts for the exponential decay of the received power level with distance. If the optimization process is performed solely based on the objective of finding more than one ground node without considering the need for maintaining communication with the control station, the UAVs that do discover multiple nodes may end up isolated if those nodes are located beyond the air-to-air range of the UAV. If a similar compensating gain is applied independently when a communication path to the control station is established, then the two objectives become conflictive and is difficult to distinguish when one will overcome the other.

A hint to a possible solution is provided in the very problem formulation: maximize coverage to as many ground nodes as possible while maintaining communication with the control station. This means that both objectives should be achieved concurrently. Therefore, the gain described in Equation (3.6) should only be applied if there is a path to the control station as defined by the

procedure explained in Section 3.2.1. Additionally, a reward factor should still be applied to the function if a UAV covers more than one ground node as defined in Equation (3.4). After including these considerations, the objective function is defined by Equation (3.7).

$$f(x_i) = G(p_e) \begin{cases} 2^{|A|-1} \sum_{j=1}^n \begin{cases} R(x_i)_j, & R(x_i)_j \geq \tau \\ 0, & R(x_i)_j < \tau \end{cases}, & |A| > 0 \\ \max_{1 \leq j \leq n} R(x_i)_j, & |A| = 0 \end{cases} \quad (3.7)$$

$$G(p_e) = \begin{cases} (0.5214l)^2 + h^2, & p_e = 1 \\ 1, & p_e = 0 \end{cases}$$

where  $p_e$  is equal to 1 if there is a path from UAV  $i$  to the control station and 0 if there is not, and  $A$  remains as defined in Equation (3.3).

Further, a linear penalty/reward approach was introduced to maintain the UAVs within a desirable range from each other. In this approach, a factor was included in the objective function to reward or penalize it according to the distance between UAVs, where excessive closeness is penalized according to proximity ranges, as illustrated in Figure 3.4.

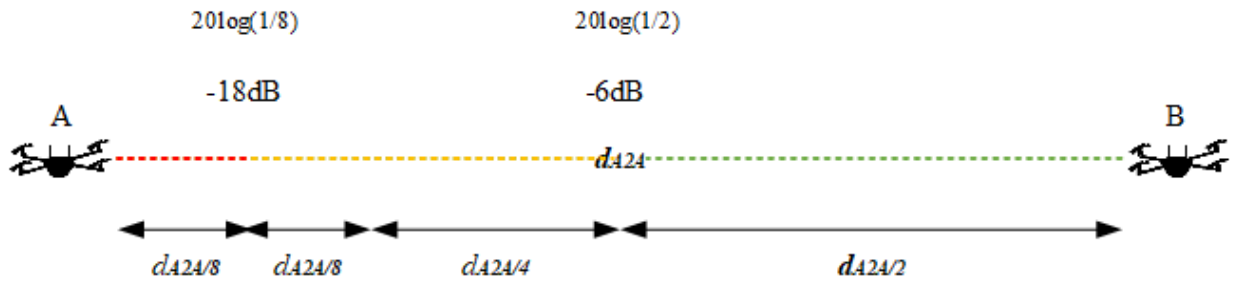


Figure 3.4: Proximity ranges between UAVs.

The proximity ranges are defined by the maximum communication distance between UAVs, which, in turn, is defined by the receiver sensitivity. From Equation (2.1), it follows that the received power level increases by approximately 6 dB—for a quadratic path loss exponent—every

time the distance between two UAVs halves. Therefore, the received power level at a fractional distance can be calculated as a function of the receiver sensitivity and the power increment in dB. Furthermore, it can be used as a measure of proximity without the need to calculate the maximum distance or determine the exact distance between UAVs, which may be convenient when exact positions are not available. The objective function, with the additional proximity avoidance consideration, is defined in Equation (3.8):

$$f'(x_i(t)) = f(x_i) \prod_{j=1}^N \begin{cases} a, & \tau \leq R'(x_i(t))_j [dBm] < \tau + 6 \text{ dB} \\ b, & \tau + 6 \text{ dB} \leq R'(x_i(t))_j [dBm] < \tau + 18 \text{ dB} \\ c, & \tau + 18 \text{ dB} \leq R'(x_i(t))_j [dBm] \\ d, & R'(x_i(t))_j [dBm] < \tau \end{cases} \quad (3.8)$$

$$a = 1, b = \frac{3}{4}, c = \frac{1}{4}, d = 1.$$

where  $N$  is the number of UAVs,  $R'(x_i(t))_j$  is the received power level from the  $j$ -th UAV at time  $t$  expressed in dBm,  $\tau$  is the receiver sensitivity, and  $f(x_i)$  remains as defined in Equation (3.7). In this case,  $f'(x_i(t))$  is a function of position and time since the UAVs are dynamic as opposed to the ground nodes.

Constant  $a$  takes a value of 1 because the range shown in green color in Figure 3.4 does not require a reward. The value of constant  $b$  is set to 0.75 as it corresponds to three fourths of the distance subject to penalty, while constant  $c$  takes a value of 0.25 to impose a penalty on being within the closest quarter of the distance subject to penalty. Constant  $d$  takes a value of 1 because if it were smaller, it would penalize not having a connection between each pair of UAVs. Furthermore, it is not necessary to penalize not having a connection between each pair of UAVs, as having a communication path to the control station is already being rewarded. In other words, constants  $a$

and  $d$  are set to 1 because forming a full mesh does not need to be rewarded and not forming it does not need to be penalized. As a result of implementing this feature, the UAVs avoid being too close to each other when they approach convergence. This does not seem to guarantee collision avoidance though, as the solution algorithm does not replace previous personal or group best values that draw a UAV to the positions where those values were recorded. Nonetheless, a velocity constraint could be applied to limit speed if the received signal power from other UAV keeps increasing over a certain time interval.

A final consideration is made to safeguard the overall connection to the control station: UAV<sub>1</sub> is rewarded or penalized correspondingly to its distance to the control station, as defined in Equation (3.9).

$$f''(x_1(t)) = \begin{cases} a' \cdot f'(x_1(t)), & \tau \leq R'(x_1)_{cs}[dBm] < \tau + 6 \text{ dB} \\ b' \cdot f'(x_1(t)), & \tau + 6 \text{ dB} \leq R'(x_1)_{cs}[dBm] < \tau + 18 \text{ dB} \\ c' \cdot f'(x_1(t)), & \tau + 18 \text{ dB} \leq R'(x_1)_{cs}[dBm] \\ d' \cdot f'(x_1(t)), & R'(x_1)_{cs}[dBm] < \tau \end{cases} \quad (3.9)$$

$$a' = (0.5214l)^2 + h^2, b' = \frac{3}{4}, c' = \frac{1}{4}, d' = \frac{1}{4}.$$

where  $f'(x_1(t))$  is the value of the objective function defined in Equation (3.8) for UAV<sub>1</sub>, and  $R'(x_1(t))_{cs}$  is the received power level at UAV<sub>1</sub> from the control station in dBm.

Constants  $b'$  and  $c'$  are established with the same consideration as in Equation (3.8). Constant  $d'$  penalizes excessive distance between UAV<sub>1</sub> and the control station in the same amount used to penalize excessive closeness. The value of  $a'$  is set high to guarantee that UAV<sub>1</sub> remains within the best range from the control station. This feature also improves convergence time for the entire swarm.

### 3.3.2 Problem Formulation

In the proposed scenario, the problem is to determine the positions of the UAVs in a way that maximizes coverage to a series of ground nodes with unknown positions, while providing a communication path from the covered nodes to a control station. Therefore, the objective function is a communications-related parameter that changes as a function of position. The selected parameter to be maximized is the aggregated received power level at  $N$  UAVs from  $n$  ground nodes. The variables of the problem are  $x_i$ ,  $i = 1, \dots, N$ , and  $j = 1, \dots, n$ , where  $x_i$  is the position of UAV  $i$ . The UAVs must remain within the search space,  $S$ , defined by a square grid of side length,  $l$ , and maximum height,  $h_{max}$ . Additionally, the received power levels can only be aggregated when they are above a certain threshold,  $\tau$ , otherwise, only the maximum received value from all ground nodes is considered. Hence, the problem can be formulated as:

$$\begin{aligned} \max_{x_i \in \mathbb{R}^3} & \begin{cases} f'(x_i(t)), & i = 2, \dots, N \\ f''(x_1(t)), & i = 1 \end{cases} \\ & \text{subject to } x_i \in S \end{aligned} \quad (3.10)$$

where  $f'(x_i(t))$  and  $f''(x_1(t))$  are defined by Equations (3.8) and (3.9), respectively.

The problem can be further characterized as a constrained, nonlinear, multiobjective optimization with multiple local solutions over a large search space. From the problem requirements and formulation, it follows that the optimization is performed at each UAV in a distributed manner. This is a convenient approach with regards to achieving autonomous flight. Moreover, this tactic is also consequent with the fact that this is a multiple-solution problem.

PSO is proposed as the optimization algorithm, due to the similarity between the FANET scenario and the nature-inspired origin of PSO.

### 3.4 Solution Using a PSO Mobility Model

The mobility model was developed starting with a basic version of the PSO algorithm for maximization problems, as described in Algorithm 3.1, where the UAVs are the particles, and all the UAVs can share their positions and received power levels among themselves, regardless of the distance between each other. A difference with the common PSO described in Chapter 2 is that here the particles are not initialized with random positions and velocities, but rather start at the same position at the corner of the grid with maximum velocity.

---

**Algorithm 3.1** Basic PSO

---

```
Create and initialize a swarm with  $N$  UAVs;
repeat while stopping criteria is false;
  for each UAV  $i = 1$  to  $N$  do
    Eval objective function  $f'(x_i(t))$ 
    //set personal best position
    if  $f'(x_i(t)) > f(bestx_i)$  then
       $bestx_i = x_i(t)$ ;
    end
    //set global best position
    if  $f'(x_i(t)) > f(bestx)$  then
       $bestx = bestx_i$ ;
    end
  end
  for each UAV  $i = 1$  to  $N$  do
    update UAV  $i$  velocity according to Equation (2.10);
    move UAV  $i$  to position  $x_i(t + \Delta t)$  according to Equation (2.11);
  end
   $t = t + \Delta t$ ;
end
```

---

The flowchart for Algorithm 3.1 is depicted in Figure 3.5.



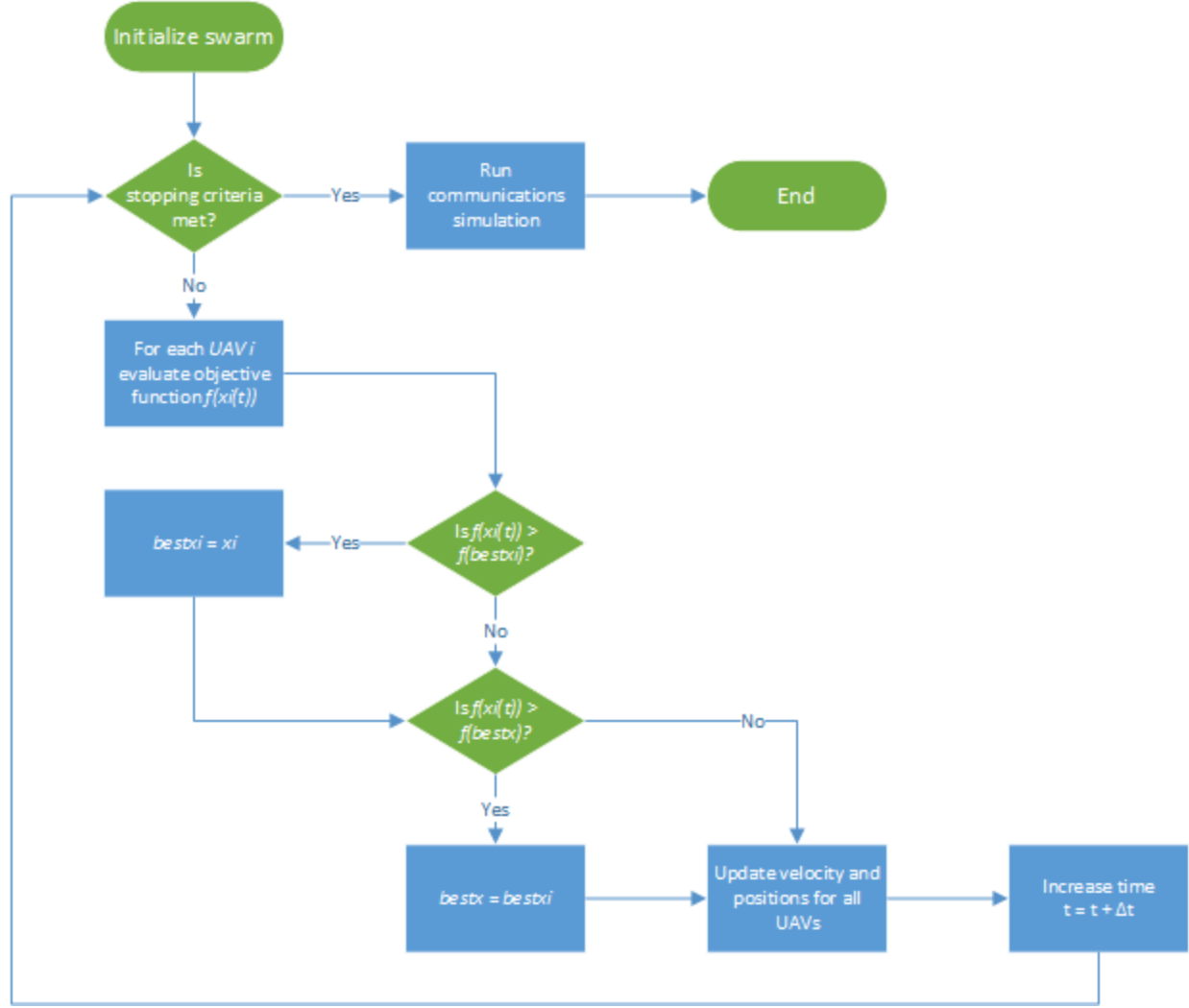


Figure 3.5: Basic PSO algorithm flowchart.

### 3.4.1 Search Space Constraint Handling

To keep the UAVs from wandering beyond the search space, which would otherwise result in increased convergence time, a modification was introduced in the algorithm to limit the UAV velocity in a given direction if the next position is outside the search space boundary in that direction. This modification, known as velocity clamping [61], is described in Algorithm 3.2.

---

#### Algorithm 3.2 PSO with bounded search space

---

Create and initialize a swarm with  $N$  UAVs;

**repeat while** *stopping criteria is false*;

**for each** UAV  $i = 1$  to  $N$  **do**

---

---

**Algorithm 3.2** PSO with bounded search space

---

```
    Eval objective function  $f'(x_i(t))$ 
    //set personal best position
    if  $f'(x_i(t)) > f(bestx_i)$  then
         $bestx_i = x_i(t)$ ;
    end
    //set global best position
    if  $f'(x_i(t)) > f(bestx)$  then
         $bestx = bestx_i$ ;
    end
end
for each UAV  $i = 1$  to  $N$  do
    calculate velocity  $v_i(t + \Delta t)$  according to Equation (2.10);
    calculate position  $x_i(t + \Delta t)$  according to Equation (2.11);
    if  $x_i(t + \Delta t)$  is outside of the search space in directions  $X, Y$ , or  $Z$  then
        update UAV  $i$  velocity making it 0 in the corresponding directions;
        move UAV  $i$  to position  $x_i(t + \Delta t)$  according to Equation (2.11);
    end
end
 $t = t + \Delta t$ ;
end
```

---

In a practical implementation, a positioning mechanism such as GPS is needed to keep the UAVs within the search space boundaries.

### 3.4.2 Kinematic Constraints Handling

The next set of modifications was introduced to keep the UAV speed below or equal to its maximum value. First, a UAV's next instant speed is clipped if its velocity magnitude is above the defined maximum. Second, a stall counter was included to determine if the search process is stagnating, as well as to adjust the inertia weight,  $w$ , accordingly. This modification is implemented as described in [80]. The stall counter increases when  $gbest$  has not improved in an iteration and decreases otherwise. Together with the stall counter, the adaptive inertia weight serves two purposes: 1) Increasing exploration when  $gbest$  keeps improving during early stages, and 2) increasing exploitation (reducing exploration) when  $gbest$  improves at later stages, which can be a sign of an optimal solution being found. Finally, the inertia weight must also be limited to keep

it from growing excessively, which might lead to low spatial resolution and increased instability, even when the maximum speed is constrained. These modifications, shown in Algorithm 3.3, help to evaluate the objective function at practical distance steps (spatial resolution) without losing potentially good values.

---

**Algorithm 3.3** PSO with maximum speed constraint

---

Create and initialize a swarm with  $N$  UAVs;

**repeat while** *stopping criteria is false*;

**for each** UAV  $i = 1$  to  $N$  **do**

    Eval objective function  $f'(x_i(t))$

    //set personal best position

**if**  $f'(x_i(t)) > f(bestx_i)$  **then**

$bestx_i = x_i(t)$ ;

**end**

    //set global best position

**if**  $f'(x_i(t)) > f(bestx)$  **then**

$bestx = bestx_i$ ;

$stallcount = \max(0, stallcount - 1)$ ;

**if**  $stallcount < 2$  **then**

$w = \min(2 * w, UAV \text{ max speed})$ ;

**end**

**if**  $stallcount > 5$  **then**

$w = w / 2$ ;

**end**

**else**

$stallcount = stallcount + 1$ ;

**end**

**end**

**for each** UAV  $i = 1$  to  $N$  **do**

    calculate velocity  $v_i(t + \Delta t)$  according to Equation (2.10);

**if**  $|UAV i \text{ velocity}| > UAV \text{ max speed}$  **then**

$UAV i \text{ velocity} = UAV i \text{ velocity} * UAV \text{ max speed} / |UAV i \text{ velocity}|$ ;

**end**

    calculate position  $x_i(t + \Delta t)$  according to Equation (2.11);

**if**  $x_i(t + \Delta t)$  is outside of the search space in directions  $X, Y$ , or  $Z$  **then**

      update UAV  $i$  velocity making it 0 in the corresponding directions;

      move UAV  $i$  to position  $x_i(t + \Delta t)$  according to Equation (2.11);

**end**

**end**

$t = t + \Delta t$ ;

**end**

---

Reducing speed to a maximum physical constraint is not mandatory in a real-world FANET implementation, as the speed would be limited naturally. However, a stall counter and an adaptive inertia weight are useful in applications with a reduced number of particles over large search

spaces. Moreover, constraining speed might be a requirement when energy efficiency is part of the problem objectives.

### 3.4.3 Dynamic Clustering

In order to account for potential loss of air-to-air links between UAVs during flight, dynamic clusters are being formed, which requires the determination of  $g_{best}$  among the neighboring UAVs. Furthermore, each UAV locally defines  $g_{best}$  without considering itself in order to avoid giving excessive weight to its own  $p_{best}$  anytime it is in fact the best within its neighborhood (see Figure 3.6).

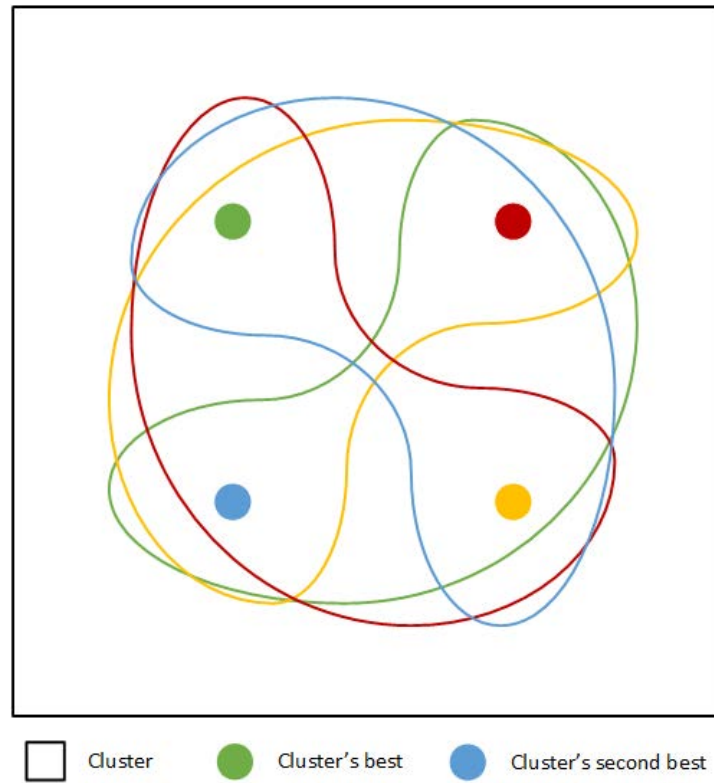


Figure 3.6: Dynamic clustering.

In the example shown in Figure 3.6, the UAV represented by the green circle is the cluster's best for the other three UAVs, while the UAV represented by the blue circle is the cluster's best for the UAV represented by the green circle. Additionally, the stall count variable and the inertia weight

are also maintained locally. Therefore, the algorithm must be updated to accommodate these circumstances, according to Algorithm 3.4.

---

**Algorithm 3.4** PSO with dynamic clustering

---

Create and initialize a swarm with  $N$  UAVs;

**repeat while** *stopping criteria is false*;

**for each** UAV  $i = 1$  to  $N$  **do**

    Eval objective function  $f'(x_i(t))$

    //set personal best position

**if**  $f'(x_i(t)) > f'(bestx_i)$  **then**

$bestx_i = x_i(t)$ ;

**end**

    //set cluster best position locally within each UVA and not considering itself

**for each** UAV  $j = 1$  to  $N$  **do**

**if**  $R'(x_i(t))_j \geq \tau$  **and**  $j \neq i$

**if**  $f'(x_j(t)) > f'(clusterbestx_i)$  **then**

$clusterbestx_i = bestx_j$ ;

$stallcount_i = \max(0, stallcount_i - 1)$ ;

**if**  $stallcount_i < 2$  **then**

$w_i = \min(2 * w_i, \text{UAV max speed})$ ;

**end**

**if**  $stallcount_i > 5$  **then**

$w_i = w_i / 2$ ;

**end**

**else**

$stallcount_i = stallcount_i + 1$ ;

**end**

**end**

**end**

**end**

**for each** UAV  $i = 1$  to  $N$  **do**

    calculate velocity  $v_i(t + \Delta t)$  according to Equation (2.10);

**if**  $|UAV i \text{ velocity}| > \text{UAV max speed}$  **then**

$UAV i \text{ velocity} = UAV i \text{ velocity} * \text{UAV max speed} / |UAV i \text{ velocity}|$ ;

**end**

    calculate position  $x_i(t + \Delta t)$  according to Equation (2.11);

**if**  $x_i(t + \Delta t)$  is outside of the search space in directions  $X$ ,  $Y$ , or  $Z$  **then**

      update UAV  $i$  velocity making it 0 in the corresponding directions;

      move UAV  $i$  to position  $x_i(t + \Delta t)$  according to Equation (2.11);

**end**

**end**

$t = t + \Delta t$ ;

**end**

---

### 3.4.4 Hybrid PSO

Inspired by the application itself, a hybrid version of the PSO algorithm (Algorithm 3.5) was devised in an attempt to increase the number of covered nodes and decrease the convergence time. In the proposed version, the grid is divided into cells depending on the number of UAVs, and a fixed waypoint trajectory is created for each UAV, from the starting point to the center of its corresponding cell at a defined height, according to Algorithm 3.6.

In the hybrid approach, during the initial phase of the flight, the UAVs evaluate the objective function and determine  $pbest$  as well as  $gbest$ , but do not start the stall count, do not update the inertia weight, and do not update their positions according to the PSO algorithm. Once all UAVs have reached the designated height for the fixed trajectory, they start the complete PSO mobility.

---

**Algorithm 3.5** Hybrid PSO with fixed initial trajectory

---

```

Create and initialize a swarm with  $N$  UAVs;
Create initial fixed waypoint trajectory;
repeat while stopping criteria is false;
  for each UAV  $i = 1$  to  $N$  do
    Eval objective function  $f'(x_i(t))$ 
    //set personal best position
    if  $f'(x_i(t)) > f'(bestx_i)$  then
       $bestx_i = x_i(t)$ ;
    end
    //set cluster best position locally within each UVA and not considering itself
    for each UAV  $j = 1$  to  $N$  do
      if  $R'(x_i(t))_j \geq \tau$  and  $j \neq i$ 
        if  $f'(x_j(t)) > f'(clusterbestx_i)$  then
           $clusterbestx_i = bestx_j$ ;
        if the fixed initial trajectory has been completed
           $stallcount_i = \max(0, stallcount_i - 1)$ ;
          if  $stallcount_i < 2$  then
             $w_i = \min(2 * w_i, \text{UAV max speed})$ ;
          end
          if  $stallcount_i > 5$  then
             $w_i = w_i / 2$ ;
          end
        end
      end
    else
      if the fixed initial trajectory has been completed

```

---



---

**Algorithm 3.5** Hybrid PSO with fixed initial trajectory

---

```
         $stallcount_i = stallcount_i + 1;$ 
    end
end
end
end
for each UAV  $i = 1$  to  $N$  do
    if the fixed initial trajectory has been completed
        calculate velocity  $v_i(t + \Delta t)$  according to Equation (2.10);
        if  $|UAV\ i\ velocity| > UAV\ max\ speed$  then
             $UAV\ i\ velocity = UAV\ i\ velocity * UAV\ max\ speed / |UAV\ i\ velocity|;$ 
        end
        calculate position  $x_i(t + \Delta t)$  according to Equation (2.11);
        if  $x_i(t + \Delta t)$  is outside of the search space in directions  $X, Y,$  or  $Z$  then
            update  $UAV\ i$  velocity making it 0 in the corresponding directions;
            move  $UAV\ i$  to position  $x_i(t + \Delta t)$  according to Equation (2.11);
        end
    else
        move  $UAV\ i$  to position  $x_i(t + \Delta t)$  according to the fixed initial trajectory;
    end
     $t = t + \Delta t;$ 
end
```

---

The initial trajectory flight for the Hybrid PSO algorithm is defined by the grid size, the number of UAVs, the maximum height of the fixed trajectory, and the UAV maximum speed ( $v_{max}$ ). Also, the fixed trajectory parameters determine the speed and direction for the first instant of the PSO-only flight.

---

**Algorithm 3.6** Create initial fixed waypoint trajectory

---

```
//set starting and ending positions
// East is X and North is Y
 $nlength = floor(sqrt(N));$ 
 $elength = ceil(N/nlength);$ 
 $FixedTRJHeight =$  fixed trajectory height;
for each UAV  $i = 1$  to  $N$  do
     $startPos_i = [-l/2, l/2, 0];$  // [X coord., Y coord., Z coord.]
end
 $i = 1;$ 
for column = 1 to  $nlength$  do
    for row = 1 to  $elength$  do
```

---

---

**Algorithm 3.6** Create initial fixed waypoint trajectory

---

```
destPosi = [-(l/2)-(l/length/2)+row*(l/length), (l/2)+(l/nlength/2)-column*(l/nlength),...  
FixedTRJHeight]; // [X coord., Y coord., Z coord.]  
i = i + 1;  
end  
end
```

---

### 3.4.5 Stopping Criteria

This parameter determines when to stop the algorithm. Common stopping criteria include reaching a maximum number of iterations, achieving a desired level of fitness, or when the improvement in fitness is below a certain threshold. Three parameters are considered as stopping criteria for the present research. The PSO algorithm stops when the objective function has not improved for a certain time, when the maximum number of iterations has been reached, or when the relative change in fitness is below a certain threshold. Two alternative functions have been developed to quantify the relative change in fitness. In the first alternative, the fitness function of UAV  $i$ , at position  $x_i$ , and time  $t$  is defined as the level of relative change in the total received power at UAV  $i$  in the last iteration with respect to the average over the previous ten iterations, as expressed in Equation (3.11):

$$\varphi_1(x_i(t)) = \frac{\left| R(x_i(t)) - \frac{\sum_{T=t-9}^t R(x_i(T))}{10} \right|}{\frac{\sum_{T=t-9}^t R(x_i(T))}{10}}. \quad (3.11)$$

The fitness threshold is reached when the maximum value of  $\varphi_1$  over all the UAVs is equal to or less than 2%. This provides a sense of the stability that the FANET has achieved regarding the received signal power.

In the second alternative, the fitness function of UAV  $i$ , at position  $x_i$ , and time  $t$  is defined as the average distance of its current and previous ten positions to the center of gravity of its previous ten positions, as expressed in Equation (3.12):

$$\begin{aligned}\varphi_2(x_i(t)) &= \frac{\sum_{T'=t-10}^t d_{x_i-CG}(T')}{11}, \\ CG &= \frac{\sum_{T=t-10}^{t-1} x_i(T)}{10}.\end{aligned}\tag{3.12}$$

The fitness threshold is reached when the maximum value of  $\varphi_2$  over all the UAVs is equal to or less than the distance a UAV travels at maximum speed during the defined time step. This alternative provides a sense of the spatial stability that the FANET has achieved and is useful with stochastic propagation models that induce a level of randomness in the received power level.

### 3.5 Simulation Parameters

In the previous sections, the objective function has been defined by Equations (3.7), (3.8) and (3.9), while the PSO mobility model has been defined by Algorithm 3.5, and Equations (3.11) and (3.12). This section covers the specific configuration parameters used in MATLAB R2022a to develop the simulation environment.

As mentioned in Section 3.2, communications start once the PSO algorithm has met the stopping criteria. The communications part of the simulation was developed at system level using the WLAN Toolbox. The term system level refers to the model ability to cover multiple links, cells, and terminals [81].

#### 3.5.1 Node configuration

The Wi-Fi nodes for UAVs and ground nodes are created using the ‘hCreateWLANNodes.m’ MATLAB function. MATLAB Wi-Fi nodes have 34 configuration parameters. The functions

‘hLoadConfigurationFull\_Int\_Traff\_6.m’ and ‘hLoadConfigurationFull\_Int\_Traff\_7.m’ (APPENDIX C) are used to configure PHY, MAC, and application traffic parameters for Wi-Fi and LoRa backhauls, respectively. Both functions are modifications of the ‘hLoadConfiguration.m’ MATLAB function. All Wi-Fi nodes are configured as mesh nodes at a frequency of 2437 MHz (Wi-Fi channel 6). A summary of all MAC and PHY configuration parameters relevant to this simulation is presented in Table 3.1. All other parameters have been set to their default values.

Table 3.1: MAC and PHY configuration parameters.

| <b>Transmit Power</b> | <b>Transmit Antenna Gain</b> | <b>Receive Antenna Gain</b> | <b>Wi-Fi Receiver Sensitivity</b> | <b>Receiver Noise Figure</b> | <b>Is Access Point</b> | <b>Is Mesh Node</b> | <b>MeshTTL</b> |
|-----------------------|------------------------------|-----------------------------|-----------------------------------|------------------------------|------------------------|---------------------|----------------|
| 15 dBm                | 1 dB                         | 0 dB                        | -82 dBm                           | 7 dBm                        | No                     | Yes                 | 31 hops        |

To simulate the LoRa backhaul for mobility performance purposes, the receiver sensitivity is set at -105 dBm, which is considered a reasonable compromise value for a 500 kHz bandwidth and a spreading factor of 5, based on the product specifications [50-52], though no specific value is provided for this configuration. The possible LoRa frequency configurations are 900 MHz, 2.4 GHz, and 400 MHz.

Regarding traffic configuration, MATLAB provides four types of WLAN application traffic: Best Effort (BE), Background (BK), Video (VI), and Voice (VO). BE traffic is defined by MATLAB as the default access category and is the one used in this model. Besides the type of traffic, traffic configurations are applied as shown in Table 3.2.

Table 3.2: Application traffic configuration.

| <b>Backhaul</b> | <b>Data Rate (kbps)</b> | <b>Packet Size [bytes]</b> | <b>Access Category</b> |
|-----------------|-------------------------|----------------------------|------------------------|
| Wi-Fi           | 100000                  | 1500                       | BE (0)                 |
| LoRa            | 62.5                    | 250                        | BE (0)                 |

The LoRa data rate can be calculated using Equation (2.9) for a 500 kHz bandwidth, a spreading factor of 5, and a 4/5 coding rate. Additionally, the traffic originates at the ground nodes and is destined for the control station. An example of traffic configuration is shown in Figure 3.7.

| SourceNode | DestinationNode | PacketSize | DataRateKbps | AccessCategory |
|------------|-----------------|------------|--------------|----------------|
| 'Node1'    | 'ControlSTA1'   | 1500       | 100000       | 0              |
| 'Node2'    | 'ControlSTA1'   | 1500       | 100000       | 0              |
| 'Node3'    | 'ControlSTA1'   | 1500       | 100000       | 0              |
| 'Node4'    | 'ControlSTA1'   | 1500       | 100000       | 0              |
| 'Node5'    | 'ControlSTA1'   | 1500       | 100000       | 0              |
| 'Node6'    | 'ControlSTA1'   | 1500       | 100000       | 0              |
| 'Node7'    | 'ControlSTA1'   | 1500       | 100000       | 0              |
| 'Node8'    | 'ControlSTA1'   | 1500       | 100000       | 0              |
| 'Node9'    | 'ControlSTA1'   | 1500       | 100000       | 0              |
| 'Node10'   | 'ControlSTA1'   | 1500       | 100000       | 0              |

Figure 3.7: Example of traffic configuration for ten ground nodes and a Wi-Fi backhaul.

### 3.5.2 Propagation Models

Three propagation models are included in the simulation: Free space, ray tracing, and log-normal. For all the propagation models, a path loss table is obtained using the function 'hCreatePathlossTableDP.m' (APPENDIX D). This function is a modification of the 'hCreatePathlossTable.m' MATLAB function.

#### A. Free Space Propagation Model

This model is implemented in the function 'hFreeSpacePathLoss.m' (APPENDIX E) as defined in Equation (2.1). The function 'hFreeSpacePathLoss.m' is a modification of the 'hTGaxResidentialPathLoss.m' MATLAB function.

#### B. Ray Tracing

This model is implemented using the 'propagationModel' MATLAB function with the following parameters:

---

```
propModel =
propagationModel("raytracing","Method","image","MaxNumReflections",1,"CoordinateSystem",
"cartesian");
```

---

### C. Log-normal

This model is implemented using the ‘propagationModel’ MATLAB function with the following parameters:

---

```
propModel = propagationModel('close-in');
propModel.PathLossExponent = 2.3;
propModel.Sigma = 0.1;
```

---

These parameters are suited to represent a rural outdoor area [34], but can be modified to represent different environments.

## 3.5.3 Mobility

The mobility model has been developed using MATLAB’s UAV Toolbox. The overall maximum height is set to 121 m (400 ft.), and the UAV maximum speed ( $v_{max}$ ) is set to 44.5 m/s (100 mph), which are defined by the Federal Aviation Administration (FAA) as the maximum altitude above ground level and the maximum groundspeed, respectively, for small unmanned aircraft [82]. The general mobility parameters are listed in Table 3.3.

Table 3.3: General mobility parameters.

| Parameter                            | Value           |
|--------------------------------------|-----------------|
| Fixed trajectory max. height         | 60 m            |
| Time step ( $\Delta t$ )             | 1 s             |
| Overall maximum height ( $h_{max}$ ) | 121 m (400 ft.) |
| UAV initial speed                    | 30 m/s          |
| UAV maximum speed ( $v_{max}$ )      | 30 m/s          |

### A. PSO Parameters

The value assigned to the PSO parameters can affect the algorithm's effectiveness and efficiency. Setting these parameters appropriately is important to ensure its performance and convergence to

an optimal solution. These parameters and the way in which they impact convergence are explained next:

- Inertia weight ( $w$ ): This parameter controls the influence of the particle's previous velocity on the current velocity. A high value of  $w$  results in increased exploration of the search space, but it also means a lower spatial resolution which may cause the algorithm to overlook positions that could produce optimal solutions. A low value of  $w$  leads to slower convergence, but it may also prevent the algorithm from getting stuck in a local optimum.
- Cognitive learning parameter ( $c_1$ ) and social learning parameter ( $c_2$ ): These parameters control the influence of the particle's personal best ( $pbest$ ) and the global best ( $gbest$ ) positions on the current velocity, respectively. A higher value of  $c_1$  increases the focus on  $pbest$ , while a higher value of  $c_2$  increases the focus on  $gbest$ .
- Number of particles ( $N$ ): This parameter determines the size of the swarm, and a higher value of  $N$  usually leads to a better exploration of the search space. However, increasing  $N$  beyond a certain threshold may also increase the computational cost for simulations, and be unattainable for real-world implementations.
- UAV maximum speed ( $v_{max}$ ): This parameter limits the maximum velocity of each particle to provide a good spatial resolution. A high value of  $v_{max}$  may lead to unstable behavior, while a low value of  $v_{max}$  may slow down the convergence. In theoretical optimization applications this value can be set to arbitrarily high values. However, in simulations and practical implementations this value is limited by aerodynamic constraints.

The relationship between the values of  $w$ ,  $c_1$ , and  $c_2$  can determine if the particles achieve convergence or if they behave erratically. According to [61,83], convergence is guaranteed if the following condition is met:

$$1 > w > \frac{1}{2}(c_1 + c_2) - 1 \geq 0. \quad (3.13)$$

As Equation (3.13) was demonstrated removing the random components, the UAVs may converge when using the stochastic components and a value of  $w$  that does not comply with this condition [61]. Given the dynamic nature of a FANET and the problem characterization provided in Section 3.3.2, a higher value of  $c_1$  and a lower value of  $c_2$  are preferred, as the opposite configuration would result in an attraction of all UAVs to a single best position. Therefore, the PSO parameters are set according to Table 3.4.

Table 3.4: PSO configuration parameters.

| Parameter                 | Value |
|---------------------------|-------|
| $w$                       | 0.95  |
| $c_1$                     | 1.35  |
| $c_2$                     | 0.01  |
| Max. number of iterations | 180   |
| Max. stall count          | 150   |

During preliminary tests, a value of -0.05 was set for  $c_2$  to cause a repelling effect between UAVs. However, a small attraction value was ultimately configured that resulted in improved path formation to avoid using a reward for keeping a path for other UAVs.  $c_1$  and  $c_2$  are fixed for the entire simulation.  $w$ , however, is only an initial value, as the adaptive, constrained inertia weight mentioned in section 3.4.2 is implemented to achieve improved convergence. Furthermore,  $w$  is independent for each UAV as per the dynamic clustering scheme described in Section 3.4.3.

Another factor that may impact PSO performance is the initial position of the particles [61]. Considering that in many of the FANET applications described in Chapter 2, it is likely that the UAVs are deployed from a single position, UAVs are initially located at one corner of the grid.



Nevertheless, different initial positions could be tested in further research. Moreover, the Hybrid PSO described in Section 3.4.4 could be considered a PSO where each UAV starts at a different position.

# **Chapter 4**

## **Results and Analysis**

As explained in Section 2.7, there are two main parts in the optimization process: (i) model development, and (ii) solution discovery. Regarding model development, the identification of variables and constraints was, to a certain extent, a straightforward process. However, the definition of an objective function that achieves multiobjective optimization proved to be more challenging. On the subject of finding a solution, setting up the PSO parameters demonstrated to be equally demanding. There exist numerous possible combinations of objective functions, PSO variations, and communication considerations. The communication details involve the propagation models and the backhaul networks to be used. These combinations were examined in a sequential manner, and the most effective alternative was chosen as input for the subsequent stage. Once the objective function has been defined and the PSO parameters have been configured, a performance analysis of the solution can be carried out in terms of performance metrics. Therefore, this chapter is organized in the following manner: Section 4.1 presents a comparison between the different objective function alternatives listed in Section 3.3. Section 4.2 shows a comparison between different PSO configurations and propagation models. The metrics used to assess the proposed FANET performance are defined in Section 4.3. Finally, three positioning algorithms are evaluated with a complete model in Section 4.4 according to the performance metrics.

### **4.1 Comparison between Different Objective Functions**

As a single objective function is proposed in the present research to achieve multiobjective optimization, multiple ways of rewarding coverage and path maintenance, as described in Section 3.3, were compared in terms of the number of iterations required for convergence or stoppage. The




model parameters presented in Table 4.1 were used for the comparison in this section, unless otherwise stated.

Table 4.1: Model configuration parameters.

| General Parameters           |   |                     |                   |
|------------------------------|---|---------------------|-------------------|
| Grid size                    | 1500 m x 1500 m   | Overall max. height | 121 m (400 ft.)   |
| Number of UAVs               | 6   | Initial speed       | 30 m/s            |
| Number of ground nodes       | 10  | Maximum speed       | 30 m/s            |
| Fixed trajectory max. height | 60 m  | Time step           | 0.5 s             |
| Radio Parameters             |   |                     |                   |
| Backhaul technology          |   | Wi-Fi               | LoRa <sup>1</sup> |
| Frequency                    |   | 2.437 GHz           | 900 MHz           |
| Tx power                     |   | 15 dBm              | 15 dBm            |
| Tx gain                      |   | 1 dB                | 1 dB              |
| Receiver sensitivity         |   | -82 dBm             | -105 dBm          |
| Rx gain                      |   | 0 dB                | 0 dB              |
| Propagation model            |   | Ray tracing         |                   |
| PSO Parameters               |   |                     |                   |
| Inertial weight ( $w$ )      | 0.95  | Max. stall count    | 150               |
| Individual weight ( $c_1$ )  | 1.35  | Max. iterations     | 180               |
| Group weight ( $c_2$ )       | 0.01  |                     |                   |
| Stopping criteria            | Mean distance to CoG, max. stall count, max. number of iterations |                     |                   |

Table 4.2 shows the meaning of the different link and trajectory plots for Figures 4.1(a)–4.18(a).

Table 4.2: Link and trajectory plot legends for Figures 4.1(a)–4.18(a).

| Link/Trajectory       | Description           | Line  |
|-----------------------|-----------------------|---|
| Air-to-air links      | Dashed line           |  |
| Air-to-ground links   | Dotted line           |  |
| No air-to-air link    | No line               |   |
| No air to ground link | No line               |   |
| Hybrid PSO trajectory | Continuous white line |  |
| PSO-only trajectory   | No line               |   |

<sup>1</sup> When applicable

### 4.1.1 Coverage Reward Only

First, the base-2 exponential gain factor was compared to the distance-based quadratic gain factor, when applied for covering more than one node, as defined by Equations (3.4) and (3.6), respectively. Figure 4.1 shows the results for the base-2 exponential gain factor, where Figure 4.1(a) shows the network topology after the stopping criteria are met, while Figure 4.1(b) shows the number of iterations required for position stabilization, which in this case is 141 and is achieved before reaching the maximum stall count or the maximum number of iterations.

Table 4.3: Alternative gains applied only for covering multiple ground nodes using Hybrid PSO (1500 m x 1500 m).

| Parameter                                      | Base-2 Exponential Gain | Distance-based Gain |
|--|-------------------------|---------------------|
| Grid size                                      | 1500 m x 1500 m         |                     |
| Mobility algorithm                             | Hybrid PSO              |                     |
| Coverage gain                                  | Equation (3.4)          | Equation (3.6)      |
| Path maintenance gain                          | ---                     | ---                 |
| Excessive closeness penalty                    | Equation (3.8)          | Equation (3.8)      |
| UAV <sub>1</sub> connection to Control Station | Equation (3.9)          | Equation (3.9)      |
| Stopping criteria reached                      | Convergence             | Max. iterations     |
| Number of iterations                           | 141                     | 180                 |

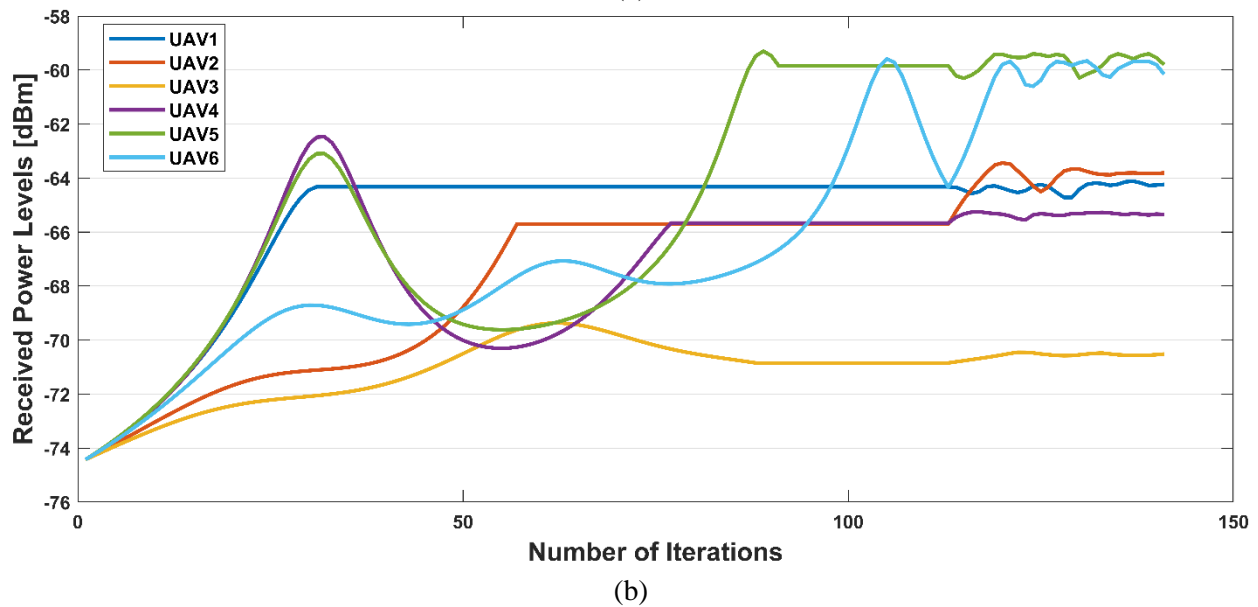
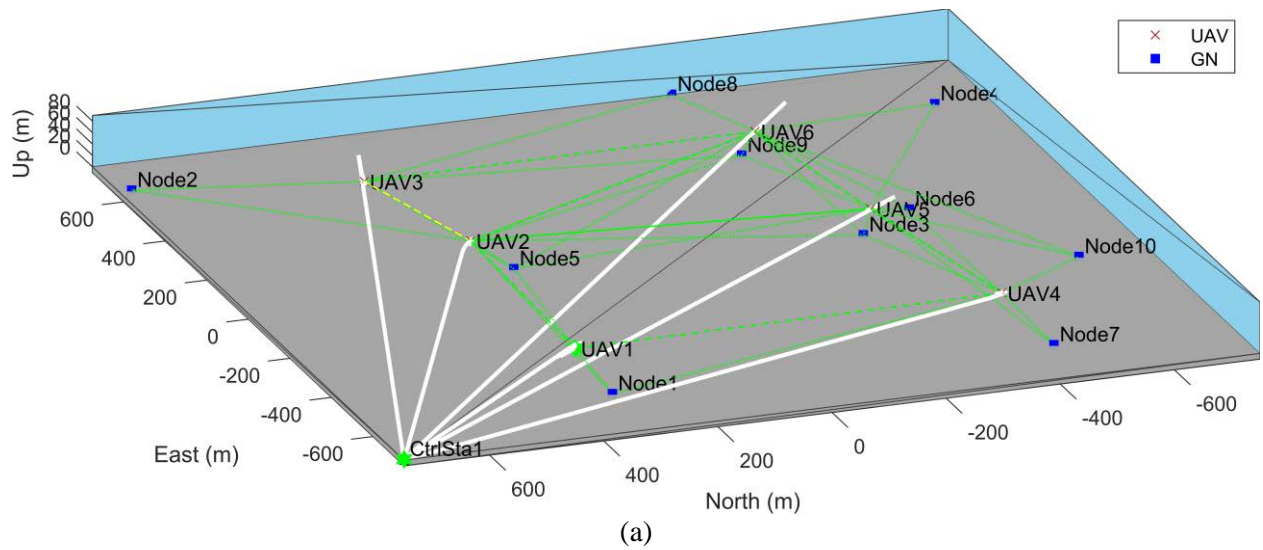
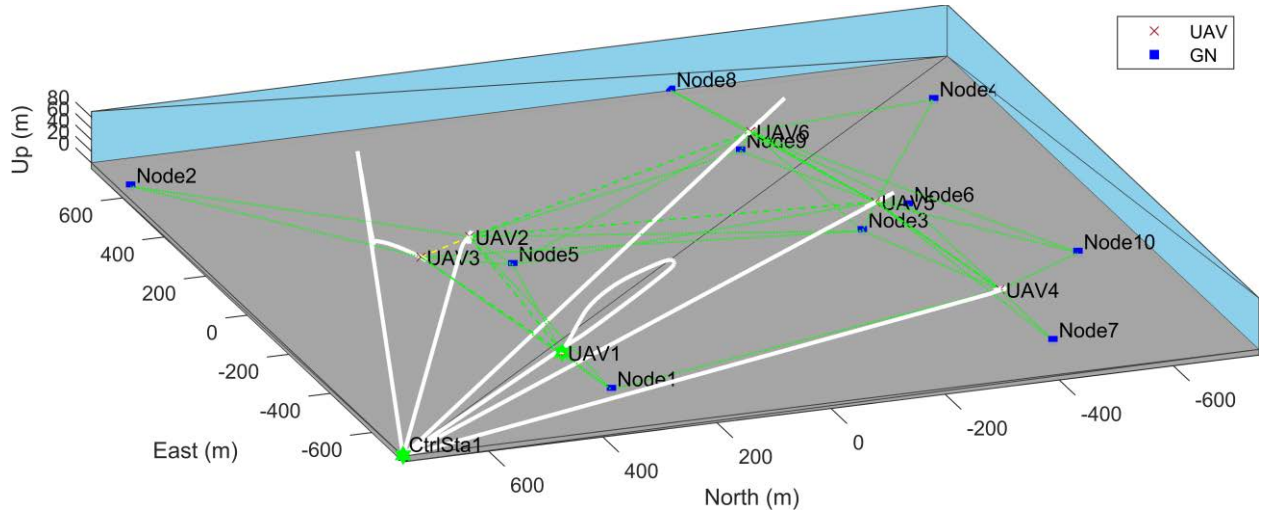
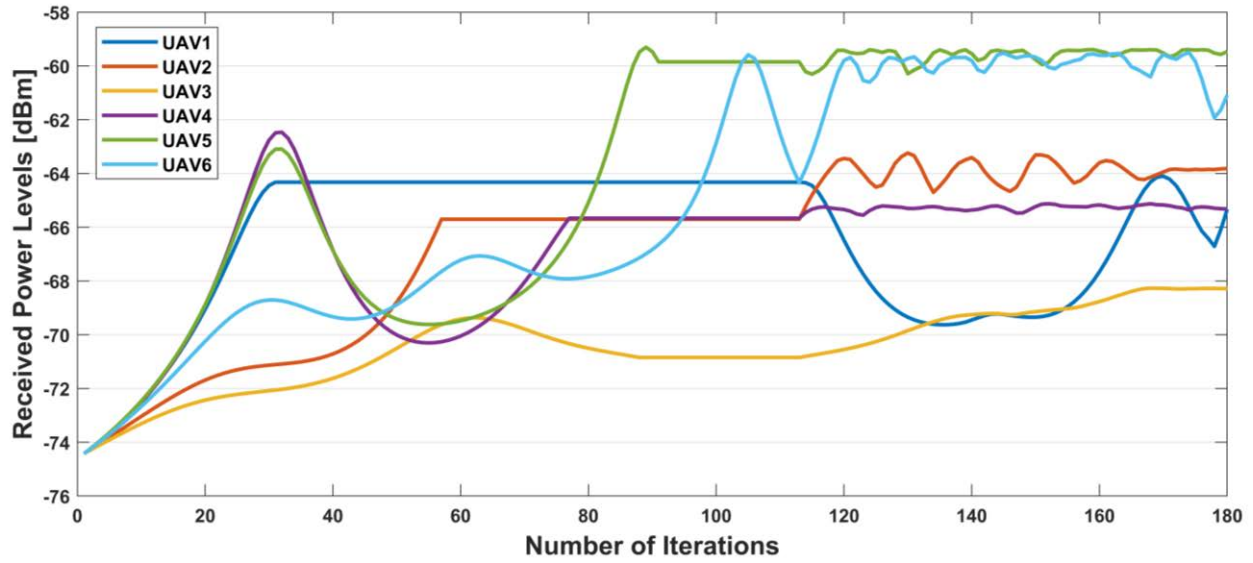


Figure 4.1: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

Figure 4.2 shows the results for the distance-based gain factor, where Figure 4.2(a) shows the network topology after the stopping criteria are met, while Figure 4.2(b) shows that the maximum number of iterations was reached before achieving convergence.



(a)



(b)

Figure 4.2: Topology and convergence results when applying a distance-based gain for covering multiple ground nodes (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

The network topology graphs show that, although both alternatives achieve coverage and a path to the control station for all ground nodes, the base-2 exponential gain factor has a better distance between UAV 2 and UAV 3. Additionally, the base-2 exponential gain factor achieves a faster and more stable convergence without applying a path maintenance reward. Hence, it might seem that the base-2 exponential gain factor is a good enough alternative compared to the distance-based gain when both are applied only for covering more than one ground node under the specified

conditions. Moreover, it might seem that multiobjective optimization has been achieved and that a path maintenance reward is not necessary, as all ground nodes have a path to the control station in option 1. Nonetheless, the objective function must perform adequately under different circumstances. Therefore, further alternatives are compared in the next section.

#### 4.1.2 Coverage and Path Maintenance Rewards

To expand the analysis from the previous section, the following alternatives were compared for different grid sizes and mobility algorithms:

1. The base-2 exponential gain factor applied for covering multiple nodes, as defined by Equation (3.4), together with a gain factor of 2 applied for having a path to the control station.
2. The distance-based quadratic gain factor applied for establishing a path to the control station, in conjunction with the application of the base-2 exponential gain factor for covering multiple nodes, as defined by Equation (3.7).

A path maintenance gain factor of 2 is applied to option 1 in order to ensure a fair comparison of results under somewhat similar conditions.

Table 4.4: Coverage and path maintenance reward alternatives using Hybrid PSO (1500 m x 1500 m).

| Parameter                                      | Base-2 Exponential Gain | Distance-based Gain |
|--|-------------------------|---------------------|
| Grid size                                      | 1500 m x 1500 m         |                     |
| Mobility algorithm                             | Hybrid PSO              |                     |
| Coverage gain                                  | Equation (3.4)          | Equation (3.7)      |
| Path maintenance gain                          | 2                       | Equation (3.7)      |
| Excessive closeness penalty                    | Equation (3.8)          | Equation (3.8)      |
| UAV <sub>1</sub> connection to Control Station | Equation (3.9)          | Equation (3.9)      |
| Stopping criteria reached                      | Convergence             | Convergence         |
| Number of iterations                           | 141                     | 141                 |
| Nodes with path to Control Station             | 10/10                   | 10/10               |

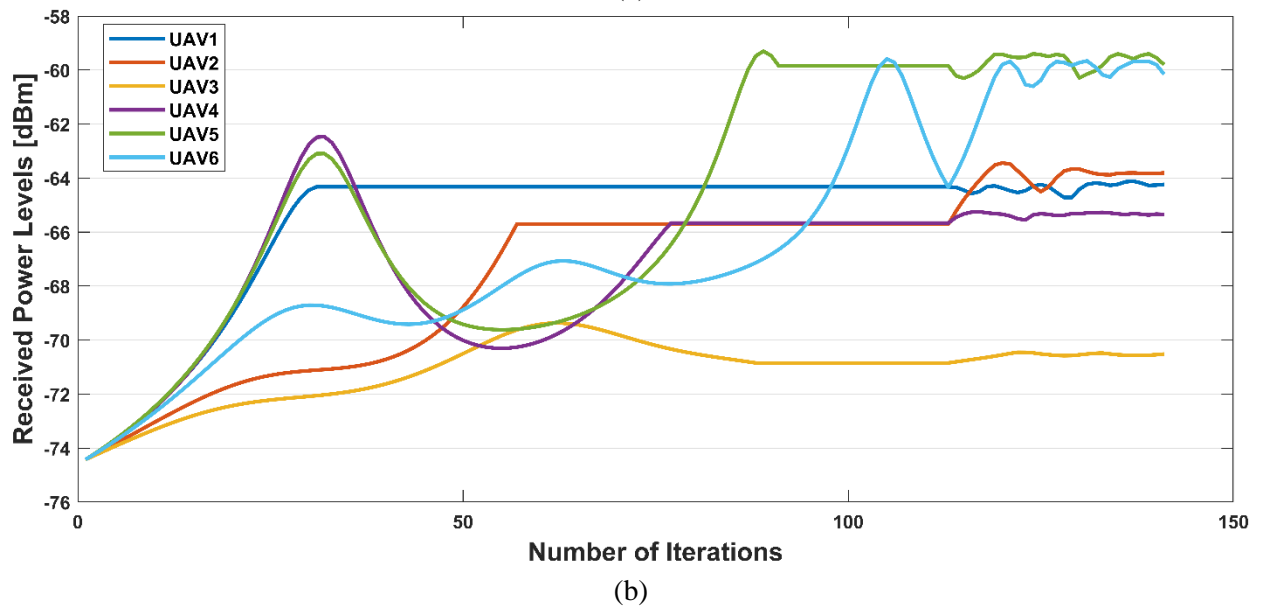
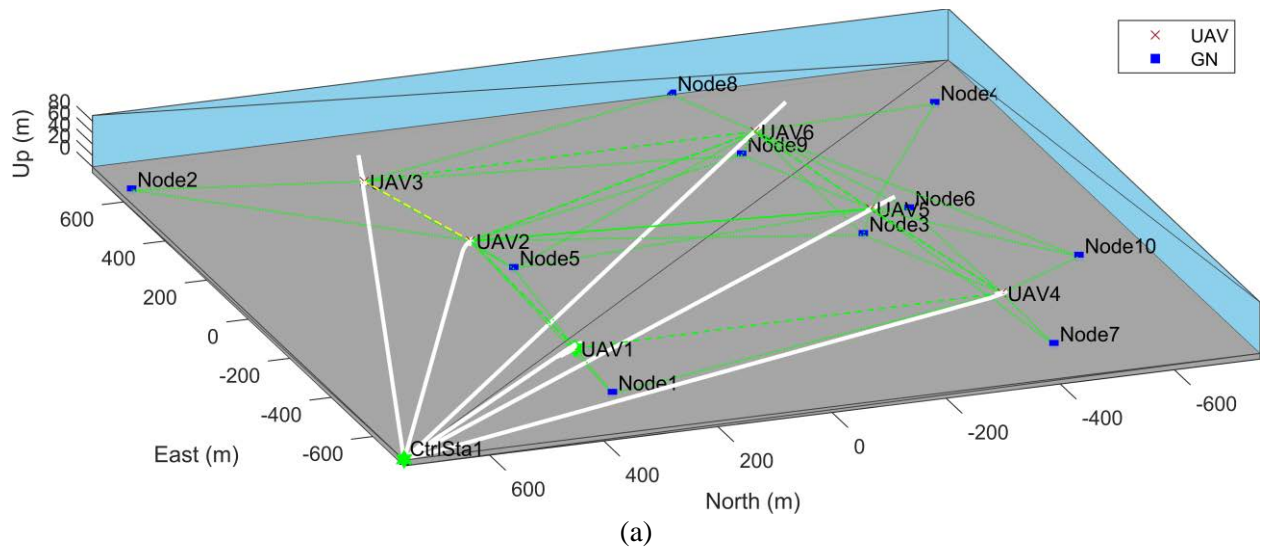
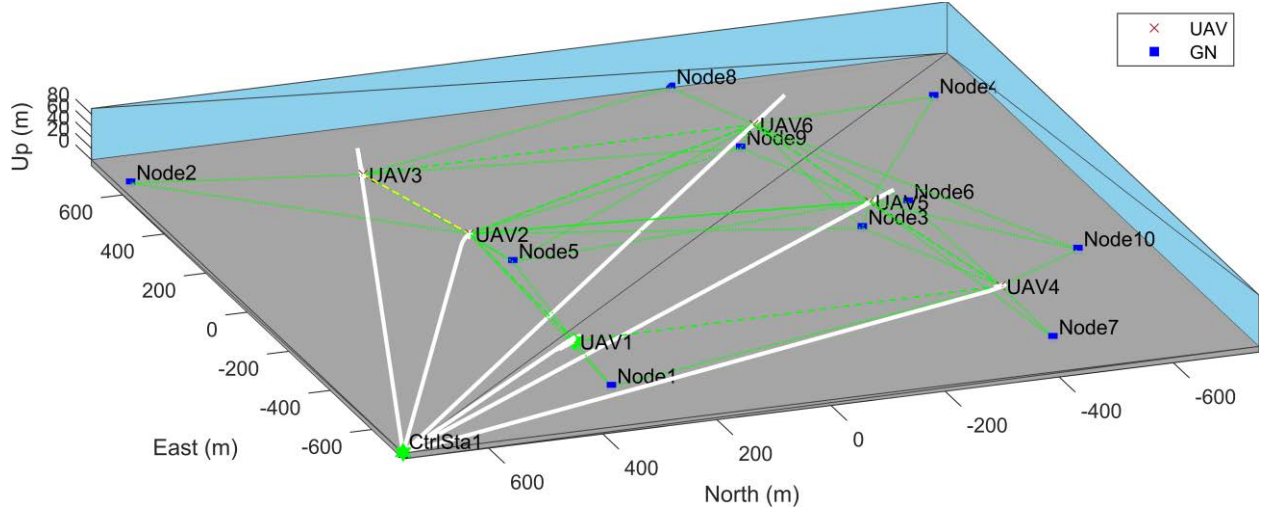
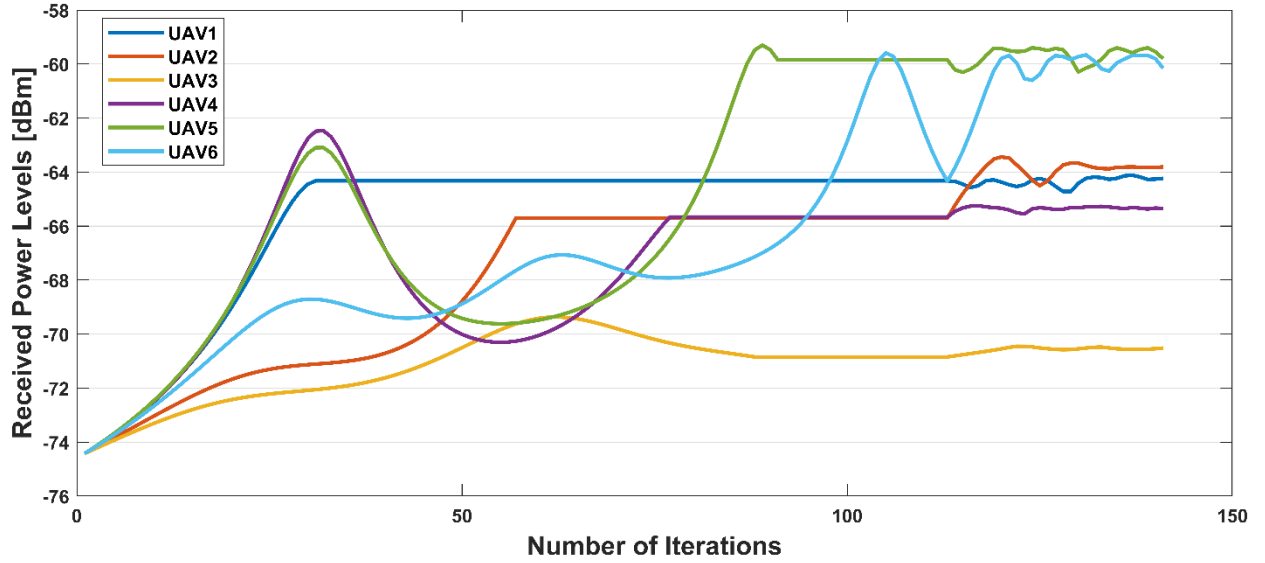


Figure 4.3: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.





(a)



(b)

Figure 4.4: Topology and convergence results when applying a distance-based gain for having a path to the control station (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

The results in Figures 4.3 and 4.4 show that both alternatives behave in the same way for the Hybrid PSO algorithm. Consequently, the Hybrid PSO algorithm was tested again increasing the grid size to 2000 m x 2000 m, and keeping all other configuration parameters as defined in Table 4.1.

Table 4.5: Coverage and path maintenance reward alternatives using Hybrid PSO (2000 m x 2000 m).

| Parameter                                      | Base-2 Exponential Gain | Distance-based Gain |
|--|-------------------------|---------------------|
| Grid size                                      | 2000 m x 2000 m         |                     |
| Mobility algorithm                             | Hybrid PSO              |                     |
| Coverage gain                                  | Equation (3.4)          | Equation (3.7)      |
| Path maintenance gain                          | 2                       | Equation (3.7)      |
| Excessive closeness penalty                    | Equation (3.8)          | Equation (3.8)      |
| UAV <sub>1</sub> connection to Control Station | Equation (3.9)          | Equation (3.9)      |
| Stopping criteria reached                      | Max. iterations         | Max. iterations     |
| Number of iterations                           | 180                     | 180                 |
| Nodes with path to Control Station             | 3/10                    | 9/10                |

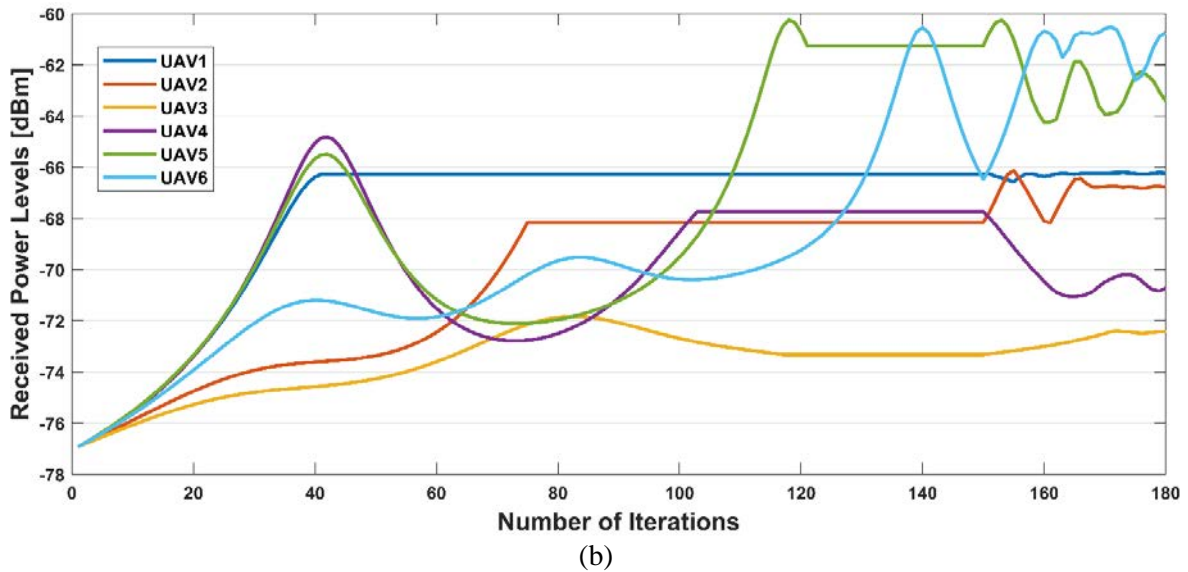
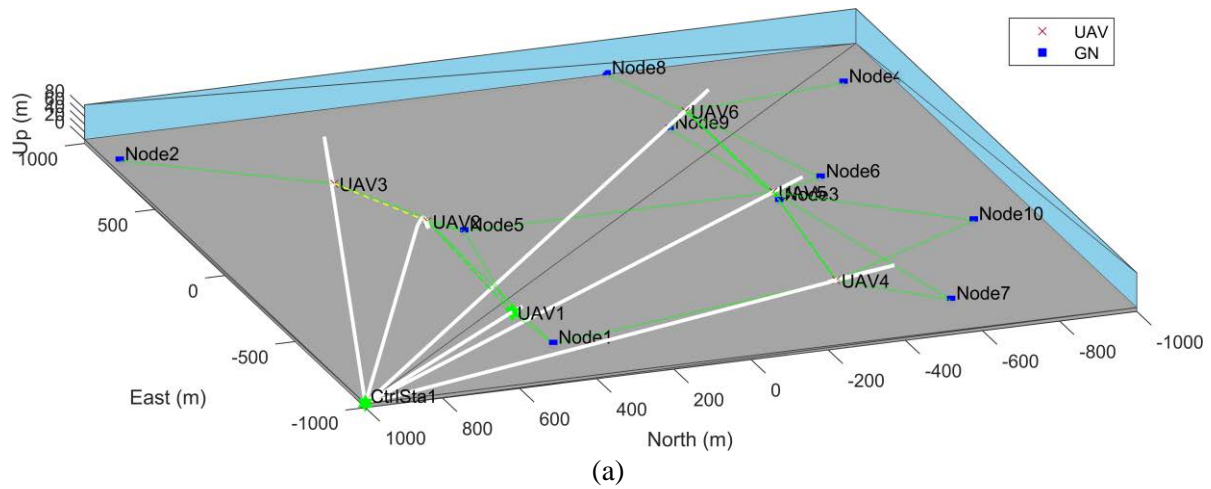


Figure 4.5: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (Hybrid PSO 2000m). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

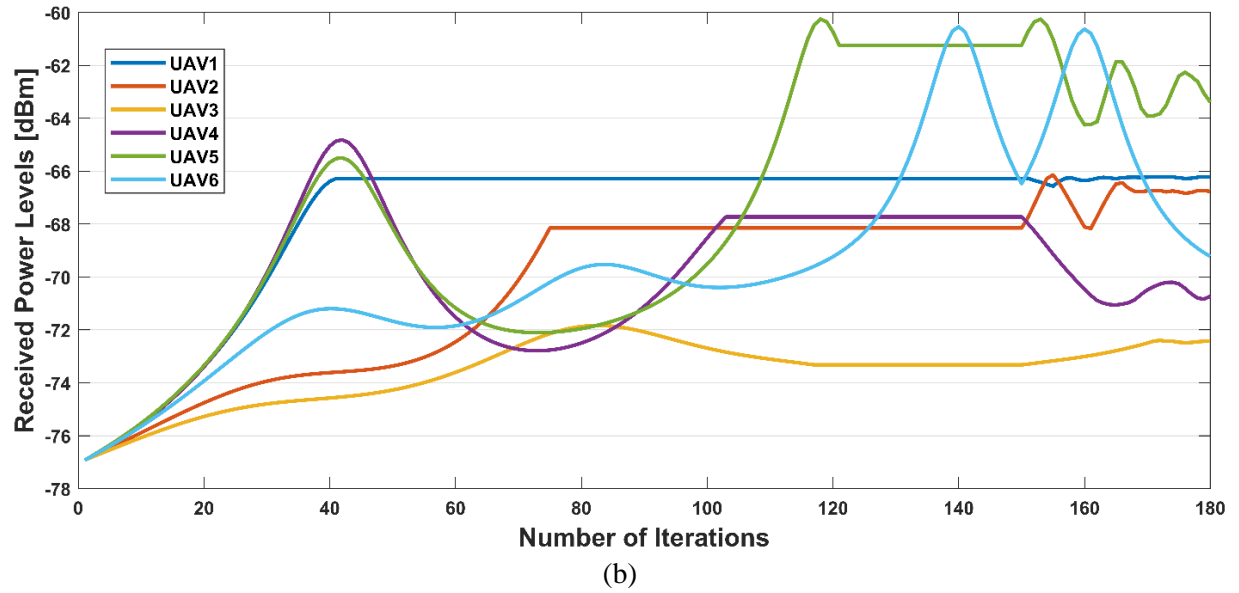
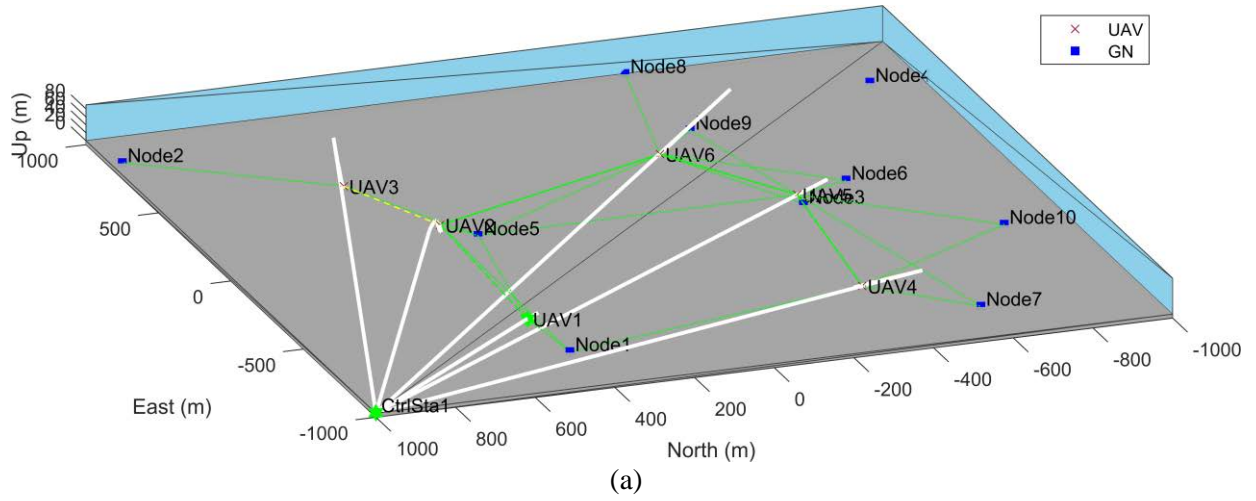
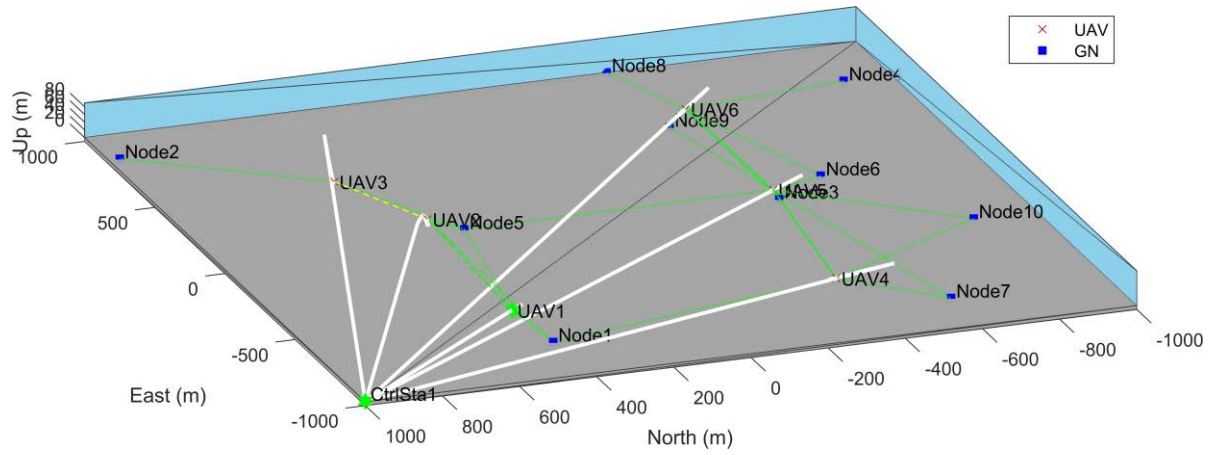


Figure 4.6: Topology and convergence results when applying a distance-based gain for having a path to the control station (Hybrid PSO 2000m). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

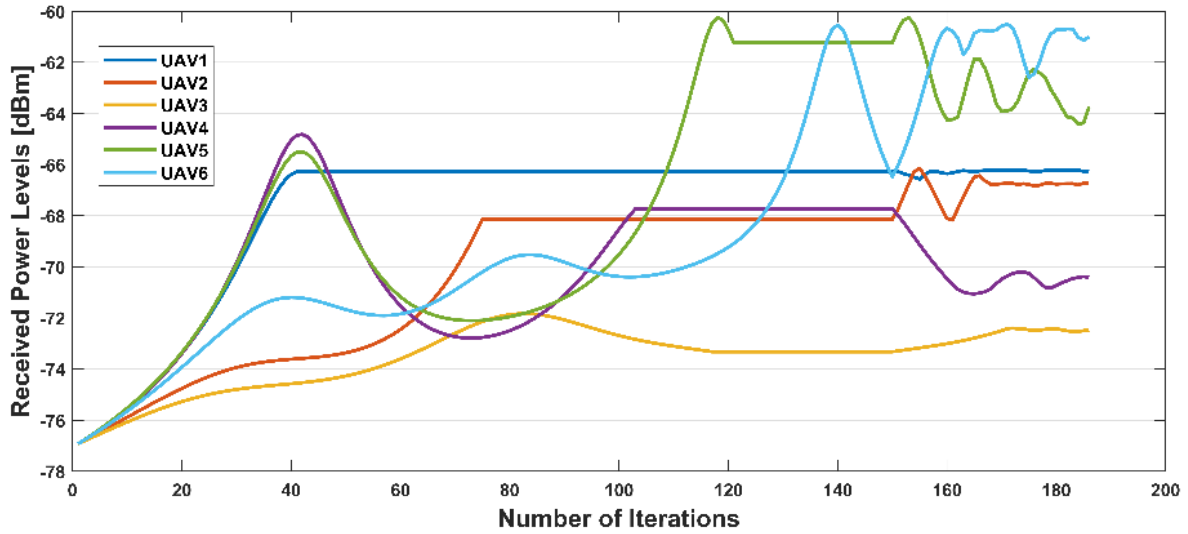
The results in Figures 4.5 and 4.6 show that the topology achieved when applying a distance-based gain for having a path to the control station outperforms that of the base-2 exponential gain for covering multiple ground nodes, since all covered ground nodes (9 out of 10) have a path to the control station. However, neither alternative achieved spatial convergence before the maximum number of iterations was reached. Thus, both alternatives were tested again, removing the maximum stall count and the maximum number of iterations from the stopping criteria, and considering the relative change in mean position only.

Table 4.6: Coverage and path maintenance reward alternatives using Hybrid PSO (2000 m x 2000 m).

| Parameter                                      | Base-2 Exponential Gain | Distance-based Gain |
|--|-------------------------|---------------------|
| Grid size                                      | 2000 m x 2000 m         |                     |
| Mobility algorithm                             | Hybrid PSO              |                     |
| Coverage gain                                  | Equation (3.4)          | Equation (3.7)      |
| Path maintenance gain                          | 2                       | Equation (3.7)      |
| Excessive closeness penalty                    | Equation (3.8)          | Equation (3.8)      |
| UAV <sub>1</sub> connection to Control Station | Equation (3.9)          | Equation (3.9)      |
| Stopping criteria reached                      | Convergence             | Convergence         |
| Number of iterations                           | 186                     | 226                 |
| Nodes with path to Control Station             | 3/10                    | 10/10               |



(a)



(b)

Figure 4.7: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (Hybrid PSO 2000m until convergence). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

It can be seen in Figure 4.7(a) that even after position convergence, a path to the control station does not exist for the UAVs at right hand side of the grid. The number of iterations in Figure 4.7(b) is 186.

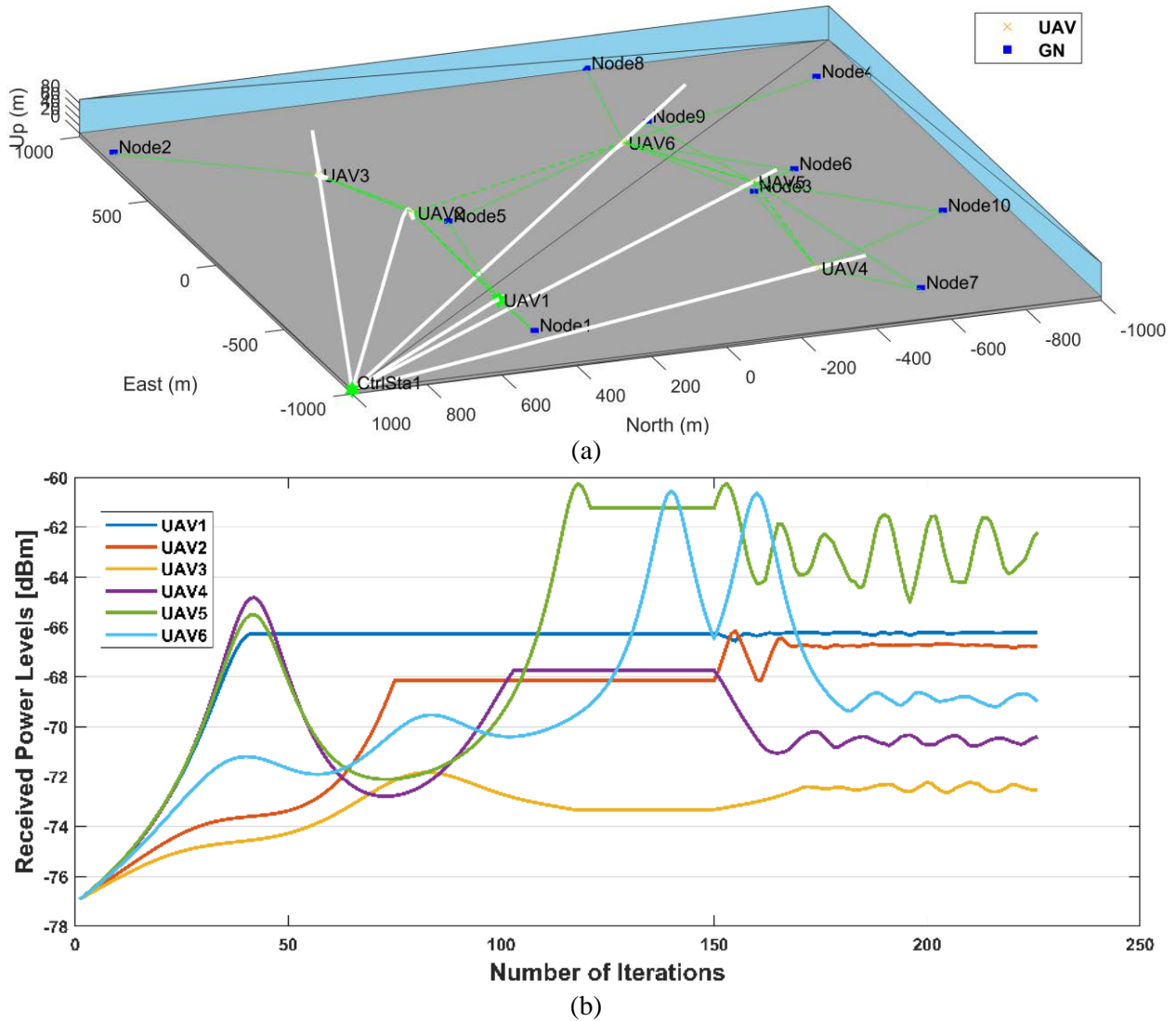


Figure 4.8: Topology and convergence results when applying a distance-based gain for having a path to the control station (Hybrid PSO 2000m until coverage). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

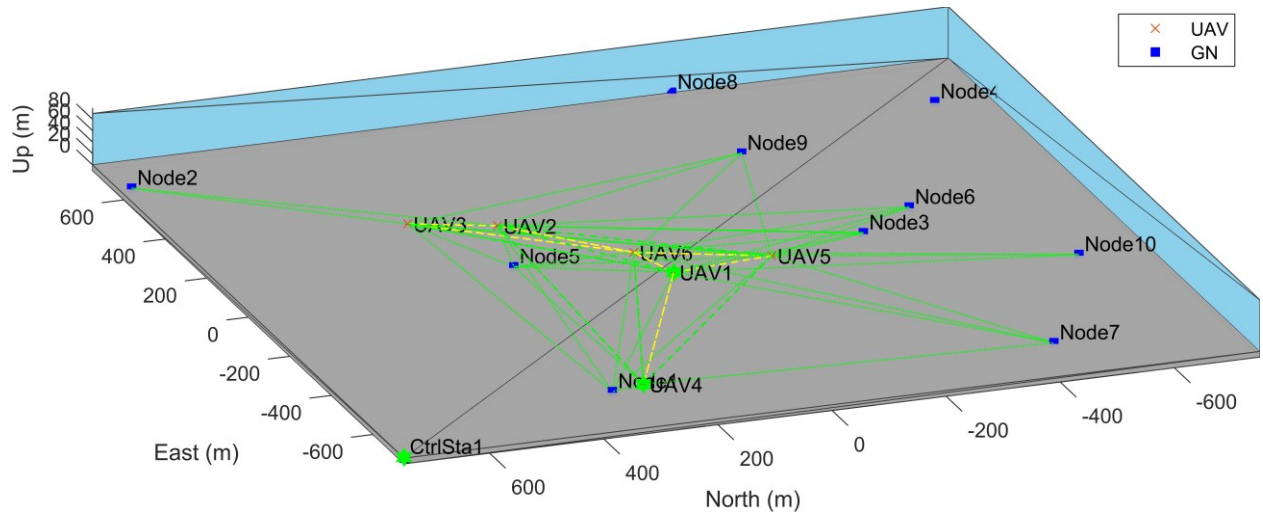
Figure 4.8(a) shows that, after position convergence, all nodes are covered and maintain a path to the control station when applying a distance-based gain. When comparing the number of iterations required for position stabilization between Figures 4.7(b) and 4.8(b), the latter requires a higher

number of iterations (226 compared to 186). In Figure 4.8(b), the number of iterations required for position stabilization is higher than when applying a base-2 exponential gain for covering multiple ground nodes (226 compared to 186 iterations). However, the importance of achieving coverage and path maintenance objectives outweighs the potential increase in convergence time. Moreover, the final positions shown in Figures 4.6(a) and 4.8(a) are virtually the same.

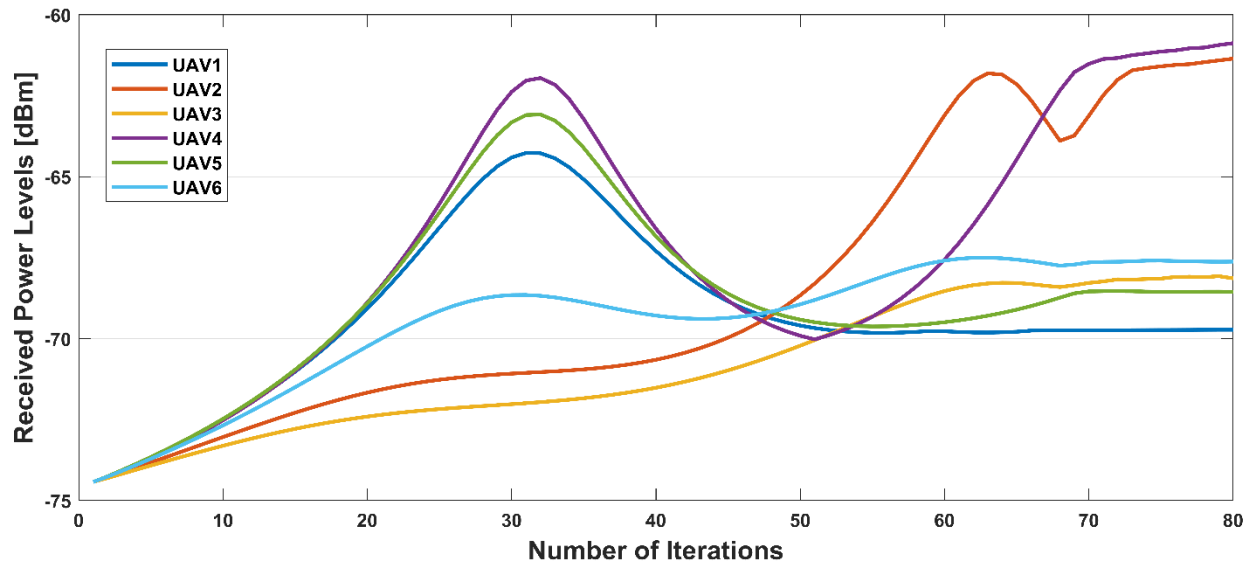
To verify that the distance-based gain for having a path to the control station outperforms the base-2 exponential gain for covering multiple ground nodes, the PSO-only algorithm was tested next with the same configuration and objective function parameters for both 1500 m x 1500 m and 2000 m x 2000 m grids, keeping all other configuration parameters as defined in Table 4.1.

Table 4.7: Coverage and path maintenance reward alternatives using PSO-only (1500 m x 1500 m).

| <b>Parameter</b>                               | <b>Base-2 Exponential Gain</b> | <b>Distance-based Gain</b> |
|--|--------------------------------|----------------------------|
| Grid size                                      | 1500 m x 1500 m                |                            |
| Mobility algorithm                             | PSO-only                       |                            |
| Coverage gain                                  | Equation (3.4)                 | Equation (3.7)             |
| Path maintenance gain                          | 2                              | Equation (3.7)             |
| Excessive closeness penalty                    | Equation (3.8)                 | Equation (3.8)             |
| UAV <sub>1</sub> connection to Control Station | Equation (3.9)                 | Equation (3.9)             |
| Stopping criteria reached                      | Convergence                    | Convergence                |
| Number of iterations                           | 80                             | 80                         |
| Nodes with path to Control Station             | 8/10                           | 8/10                       |



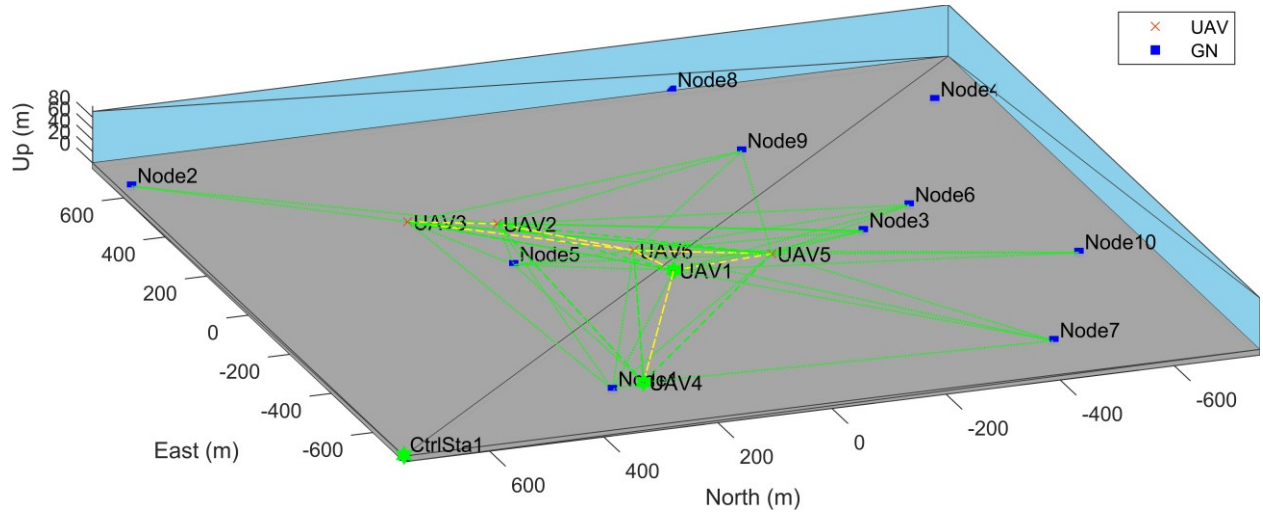
(a)



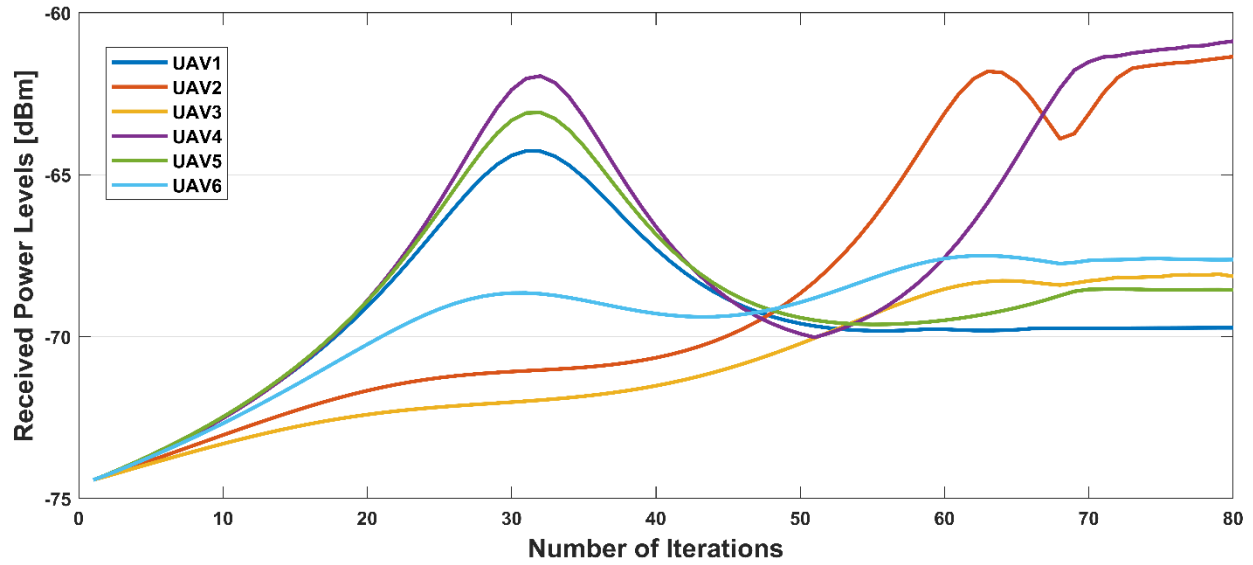
(b)

Figure 4.9: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (PSO-only). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.





(a)



(b)

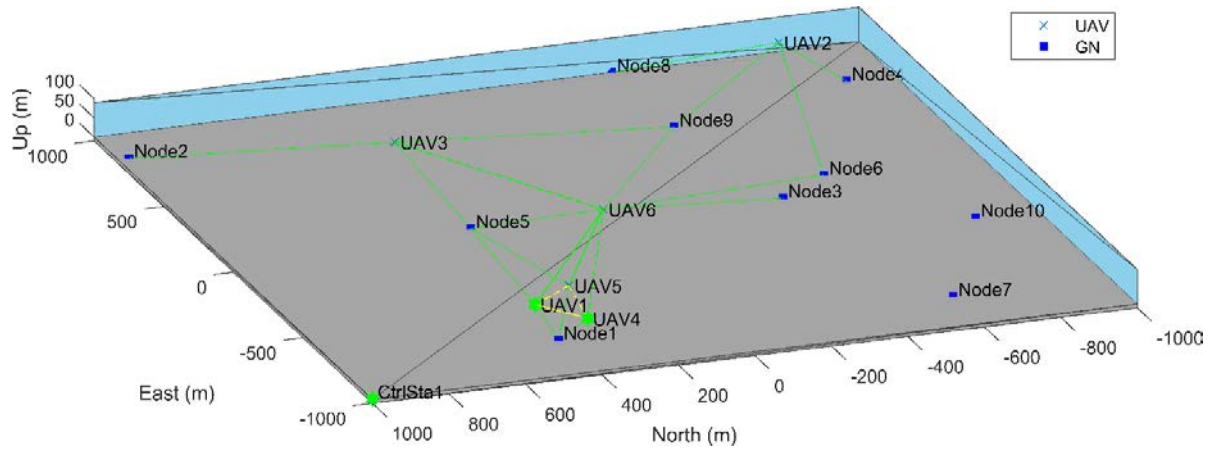
Figure 4.10: Topology and convergence results when applying a distance-based gain for having a path to the control station (PSO-only). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

The results in Figures 4.9 and 4.10 show that both gain alternatives behave in the same way for a 1500 x 1500 m grid with the PSO-only algorithm as well. Therefore, the PSO-only algorithm was tested next for a 2000 x 2000 m grid with the same configuration and objective function parameters.

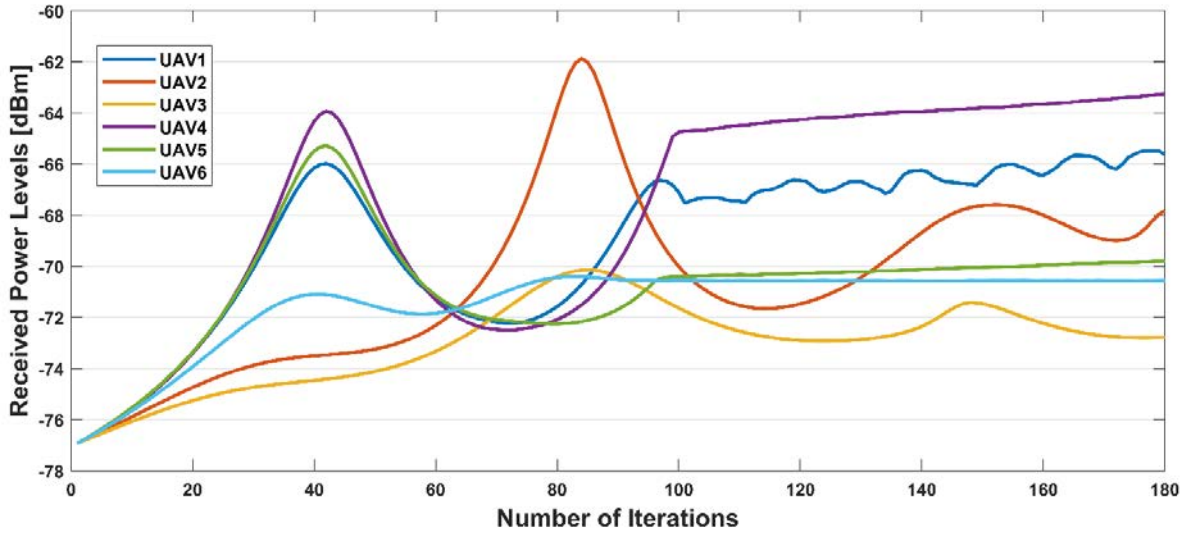


Table 4.8: Coverage and path maintenance reward alternatives using PSO-only (2000 m x 2000 m).

| Parameter                                      | Base-2 Exponential Gain | Distance-based Gain |
|--|-------------------------|---------------------|
| Grid size                                      | 2000 m x 2000 m         |                     |
| Mobility algorithm                             | PSO-only                |                     |
| Coverage gain                                  | Equation (3.4)          | Equation (3.7)      |
| Path maintenance gain                          | 2                       | Equation (3.7)      |
| Excessive closeness penalty                    | Equation (3.8)          | Equation (3.8)      |
| UAV <sub>1</sub> connection to Control Station | Equation (3.9)          | Equation (3.9)      |
| Stopping criteria reached                      | Max. Iterations         | Convergence         |
| Number of iterations                           | 180                     | 110                 |
| Nodes with path to Control Station             | 6/10                    | 3/10                |



(a)



(b)

Figure 4.11: Topology and convergence results when applying a base-2 exponential gain for covering multiple ground nodes (PSO-only 2000m). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

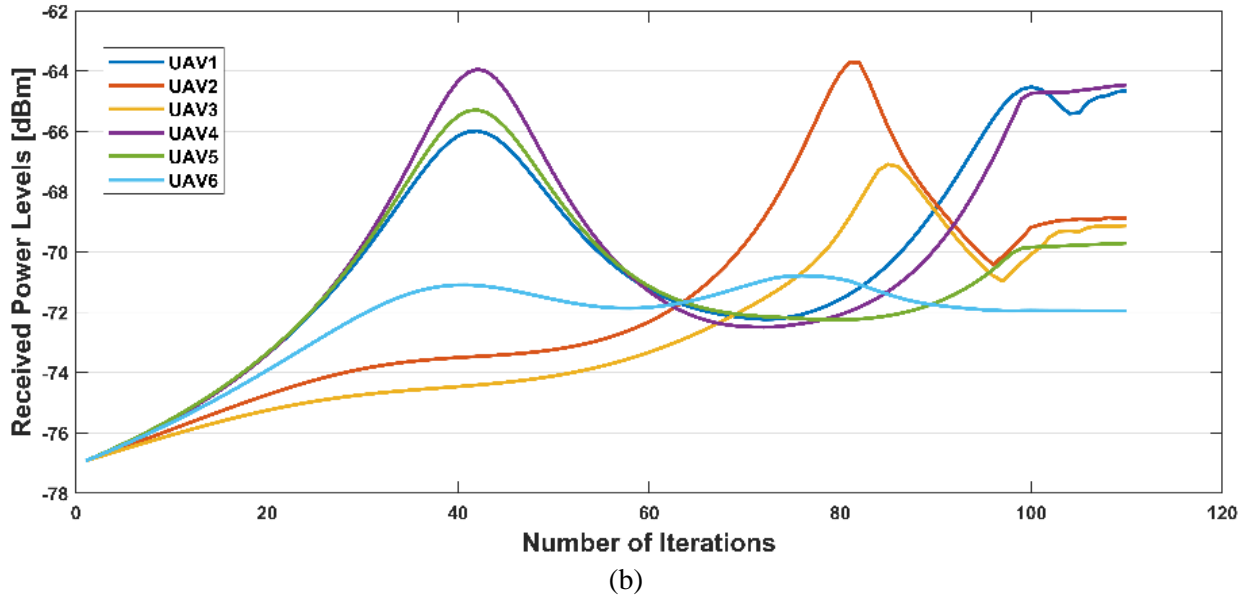
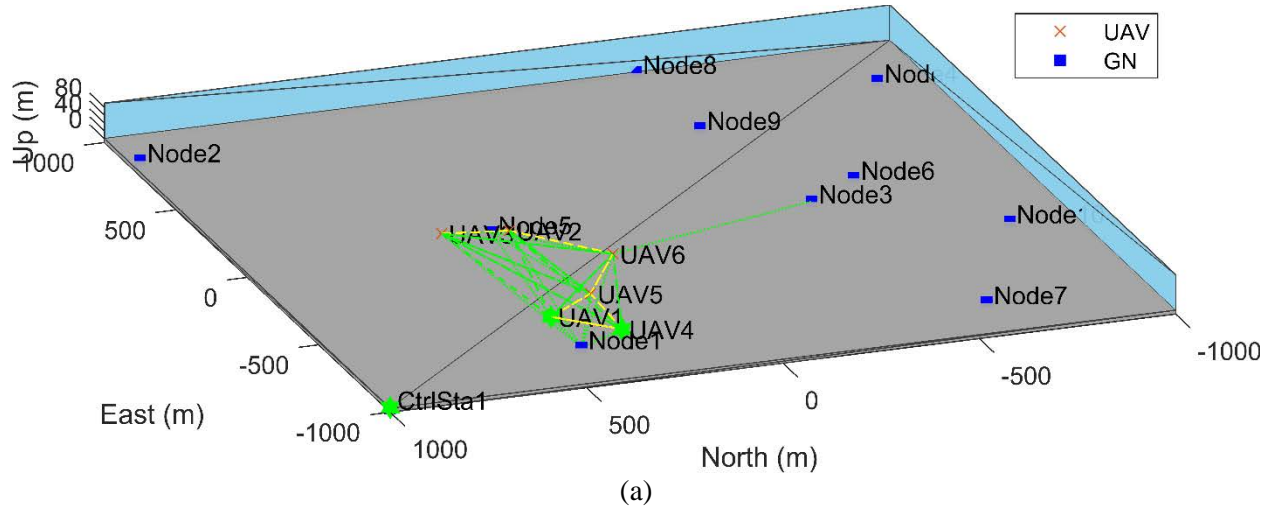


Figure 4.12: Topology and convergence results when applying a distance-based gain for having a path to the control station (PSO-only 2000m). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

Figure 4.12(b) shows that by using a distance-based gain, convergence was achieved before reaching the maximum stall count or the maximum number of iterations, and, though only three ground nodes are covered, all UAVs have a path to the control station—as well as the covered nodes. Moreover, applying a base-2 exponential gain for covering multiple ground nodes in this configuration did not achieve convergence after more than 18,000 iterations. This shows that the

distance-based gain adapts better to changes in grid size for both Hybrid PSO and PSO-only algorithms. Therefore, as explained in Section 3.3, the objective function is defined by Equations (3.7), (3.8) and (3.9) for the comparison between different PSO configurations. The results for the objective function analysis are summarized in Tables 4.9 and 4.10:

Table 4.9: Summary of results for the objective function analysis for the Hybrid PSO.

| Parameter                                      | Base-2 Exponential Gain |             | Distance-based Gain |             |
|--|-------------------------|-------------|---------------------|-------------|
| Mobility algorithm                             | Hybrid PSO              |             |                     |             |
| Coverage gain                                  | Equation (3.4)          |             | Equation (3.7)      |             |
| Path maintenance gain                          | 2                       |             | Equation (3.7)      |             |
| Excessive closeness penalty                    | Equation (3.8)          |             | Equation (3.8)      |             |
| UAV <sub>1</sub> connection to Control Station | Equation (3.9)          |             | Equation (3.9)      |             |
| Grid size                                      | 1500 m                  | 2000 m      | 1500 m              | 2000 m      |
| Stopping criteria reached                      | Convergence             | Convergence | Convergence         | Convergence |
| Number of iterations                           | 141                     | 186         | 141                 | 226         |
| Nodes with path to Control Station             | 10/10                   | 3/10        | 10/10               | 10/10       |
| Figure   | Figure 4.3              | Figure 4.7  | Figure 4.4          | Figure 4.8  |

Table 4.10: Summary of results for the objective function analysis for PSO-only.

| Parameter                                      | Base-2 Exponential Gain |                 | Distance-based Gain |             |
|--|-------------------------|-----------------|---------------------|-------------|
| Mobility algorithm                             | PSO-only                |                 |                     |             |
| Coverage gain                                  | Equation (3.4)          |                 | Equation (3.7)      |             |
| Path maintenance gain                          | 2                       |                 | Equation (3.7)      |             |
| Excessive closeness penalty                    | Equation (3.8)          |                 | Equation (3.8)      |             |
| UAV <sub>1</sub> connection to Control Station | Equation (3.9)          |                 | Equation (3.9)      |             |
| Grid size                                      | 1500 m                  | 2000 m          | 1500 m              | 2000 m      |
| Stopping criteria reached                      | Convergence             | Max. Iterations | Convergence         | Convergence |
| Number of iterations                           | 80                      | 180             | 80                  | 110         |
| Nodes with path to Control Station             | 8/10                    | 6/10            | 8/10                | 3/10        |
| Figure   | Figure 4.9              | Figure 4.11     | Figure 4.10         | Figure 4.12 |

## 4.2 Comparison between Different PSO Configurations

Once the objective function has been defined, the proposed solution algorithm, as described in Section 3.4, must be configured adequately. In this section, the performance of different PSO configurations is compared in terms of the number of iterations required for convergence or

stoppage, according to the general configuration parameters from Table 4.1, unless otherwise stated.

#### 4.2.1 Adaptive Inertia Weight vs. Guaranteed Convergence Parameters

The performance of the adaptive inertia weight mechanism described in Section 3.4.2 is compared to that of PSO parameters established for guaranteed convergence according to Equation (3.13).

The values set in each case for  $w$ ,  $c_1$ , and  $c_2$  are shown in Table 4.11.

Table 4.11: Adaptive inertia weight vs. guaranteed convergence parameters.

| Parameter                                      | Adaptive Inertia Weight | Guaranteed Convergence |
|--|-------------------------|------------------------|
| Grid size                                      | 1500 m x 1500 m         |                        |
| Mobility algorithm                             | Hybrid PSO              |                        |
| Coverage gain                                  | Equation (3.7)          |                        |
| Path maintenance gain                          | Equation (3.7)          |                        |
| Excessive closeness penalty                    | Equation (3.8)          |                        |
| UAV <sub>1</sub> connection to Control Station | Equation (3.9)          |                        |
| Inertial weight ( $w$ )                        | 0.95                    | 0.95                   |
| Individual weight ( $c_1$ )                    | 1.35                    | 1.85                   |
| Group weight ( $c_2$ )                         | 0.01                    | 0.25                   |
| Stopping criteria reached                      | Convergence             | Convergence            |
| Number of iterations                           | 141                     | 175                    |

As shown in Figures 4.13(a) and 4.14(a), the network topology after convergence is similar for both alternatives, with the adaptive inertia weight providing a slightly better topology when considering the distance between UAV<sub>2</sub> and UAV<sub>3</sub>.

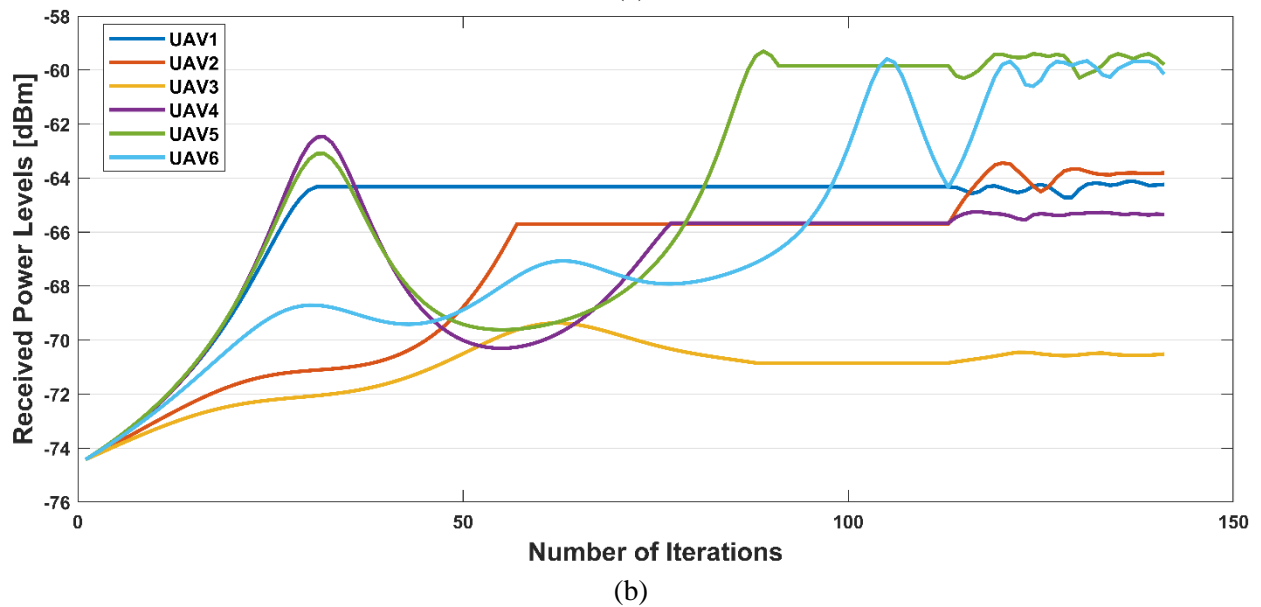
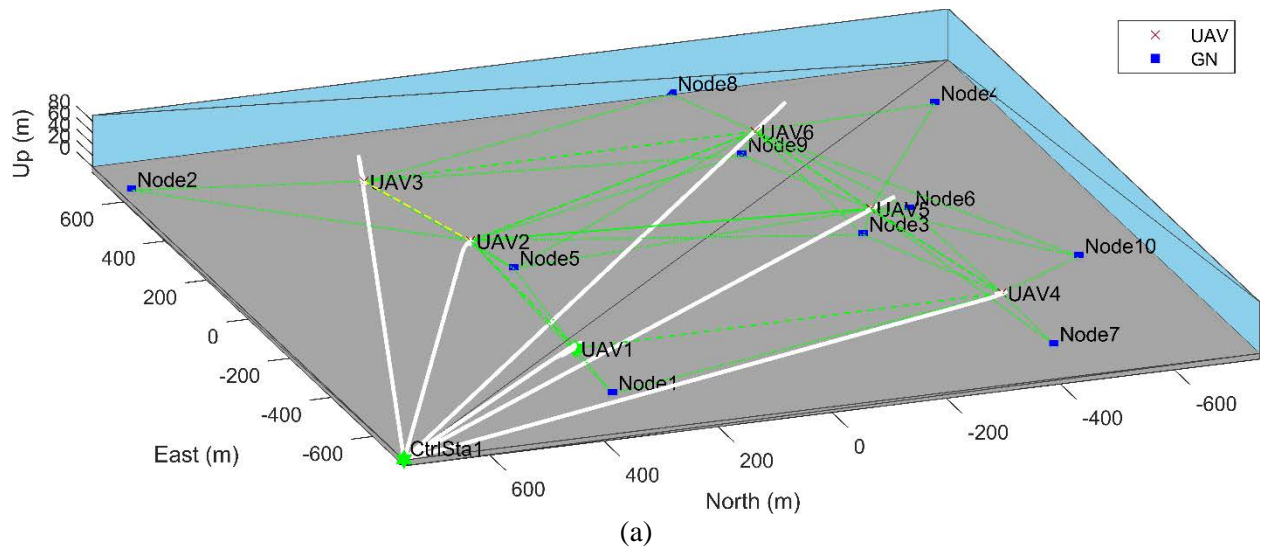
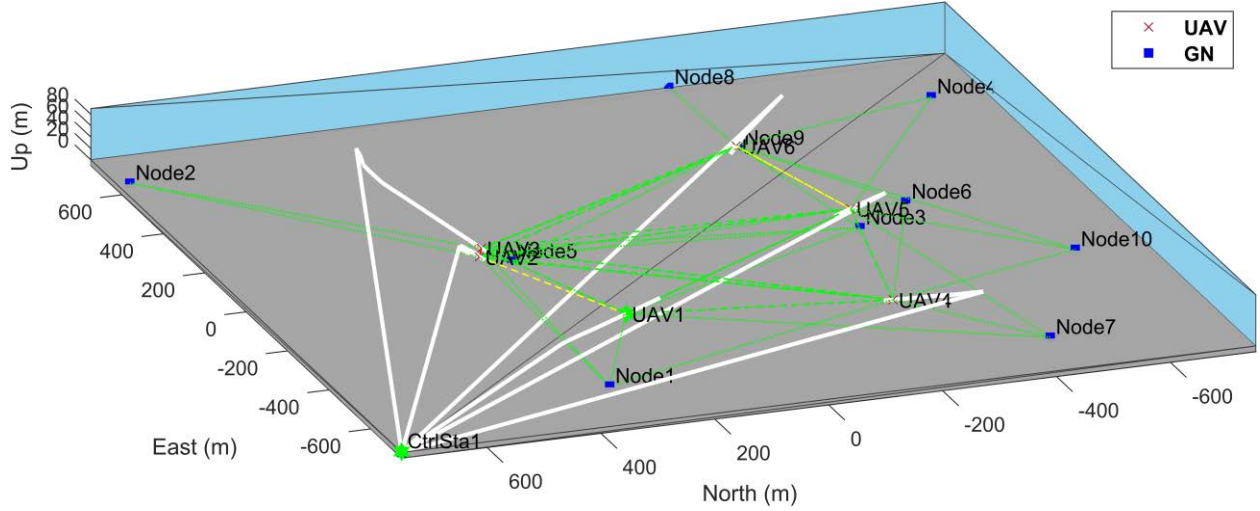
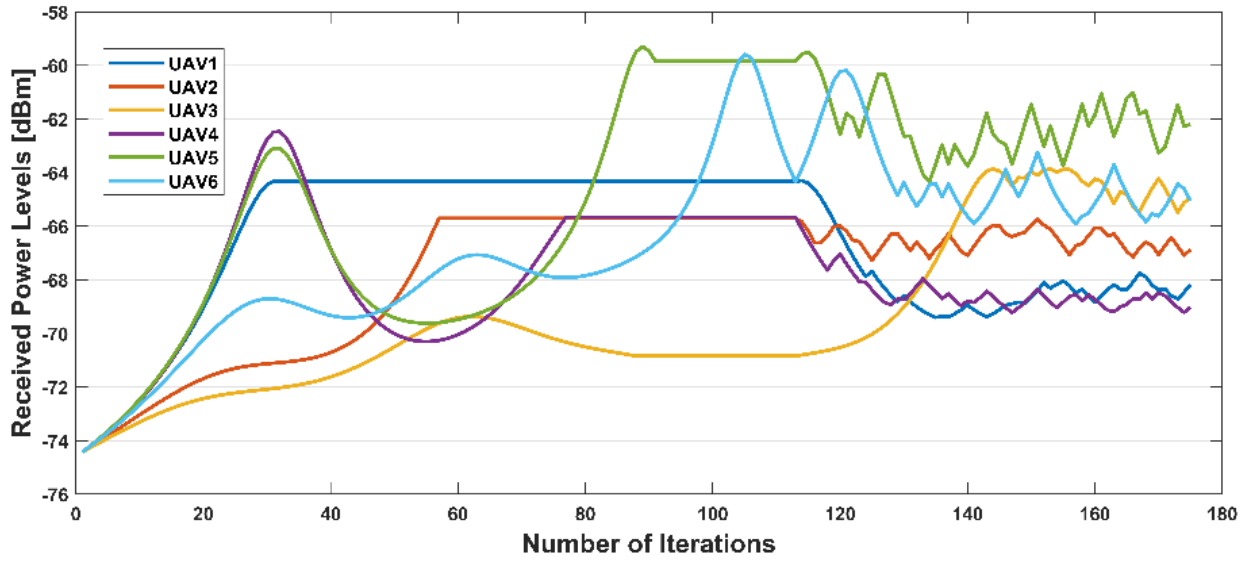


Figure 4.13: Topology and convergence results when using adaptive inertia weight (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.



(a)



(b)

Figure 4.14: Topology and convergence results when using guaranteed convergence PSO parameters (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

Additionally, a more stable convergence is achieved sooner when using an adaptive inertia weight (141 vs. 175 iterations), as can be seen in Figures 4.13(b) and 4.14(b). Therefore, the use of the adaptive inertia weight with  $w = 0.95$ ,  $c_1 = 1.35$ , and  $c_2 = 0.01$  is preferred over the values for guaranteed convergence and kept for further testing.

### 4.2.2 Propagation Model Alternatives

The implemented propagation models allow to simulate radio coverage in different scenarios. The free space model is a theoretical concept that serves as a reference for evaluating radio waves propagation through free space. It assumes there is no presence of obstacles or other forms of interference, making it a model that only applies under ideal conditions. The ray tracing model performs similarly to the free space model but considers reflections from the ground and from other UAVs; therefore, is suited for the ideal geometry of the proposed scenario. Moreover, it is ready to account for terrain and buildings if such maps are included in future research. The log-normal model considers stochastic shadowing effects and is configured for a rural outdoor area (path loss exponent equal to 2.2 and sigma equal to 0.1), which provides a reference of how the simulation would perform on irregular terrain with scattered obstructions. Therefore, ray tracing and log-normal models will be used for the performance analysis.

Table 4.12: Propagation model alternatives using adaptive inertia weight.

| Parameter                                      | Ray tracing     | Log-normal  |
|--|-----------------|-------------|
| Grid size                                      | 1500 m x 1500 m |             |
| Mobility algorithm                             | Hybrid PSO      |             |
| Coverage gain                                  | Equation (3.7)  |             |
| Path maintenance gain                          | Equation (3.7)  |             |
| Excessive closeness penalty                    | Equation (3.8)  |             |
| UAV <sub>1</sub> connection to Control Station | Equation (3.9)  |             |
| Stopping criteria reached                      | Convergence     | Convergence |
| Number of iterations                           | 141             | 135         |

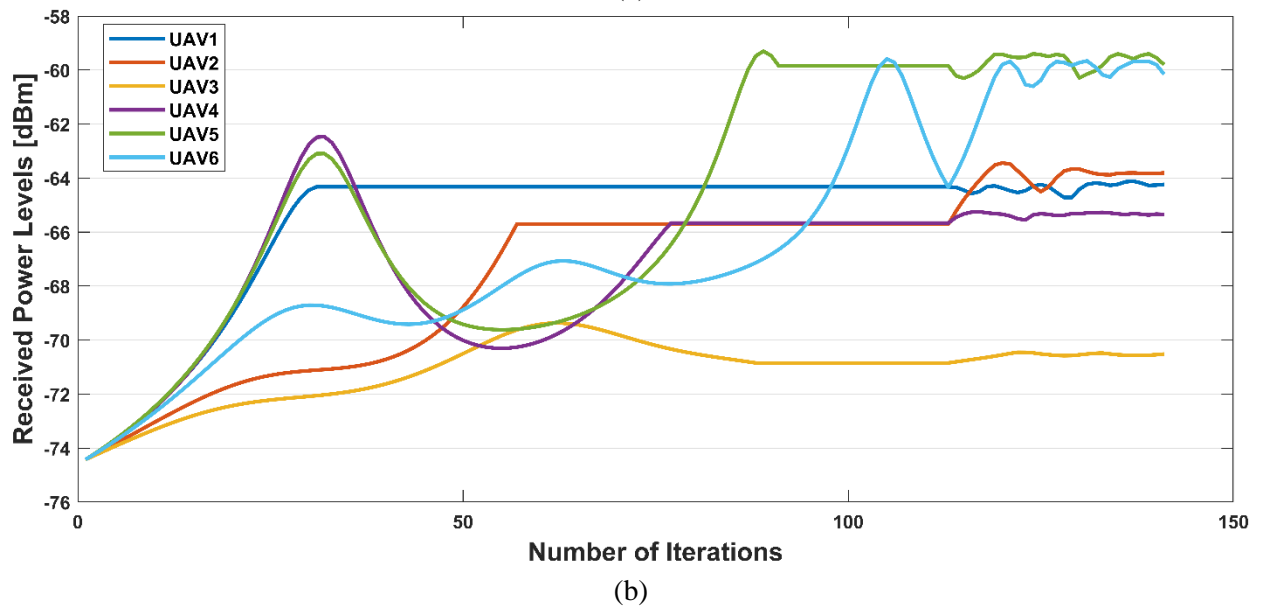
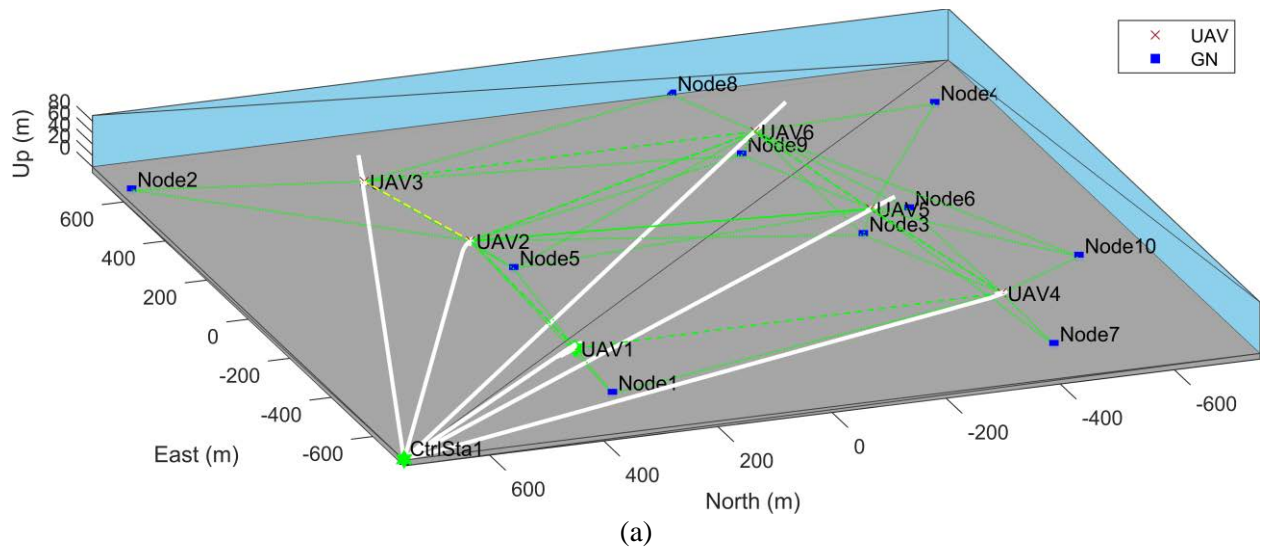


Figure 4.15: Topology and convergence results when using a ray tracing propagation model (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.



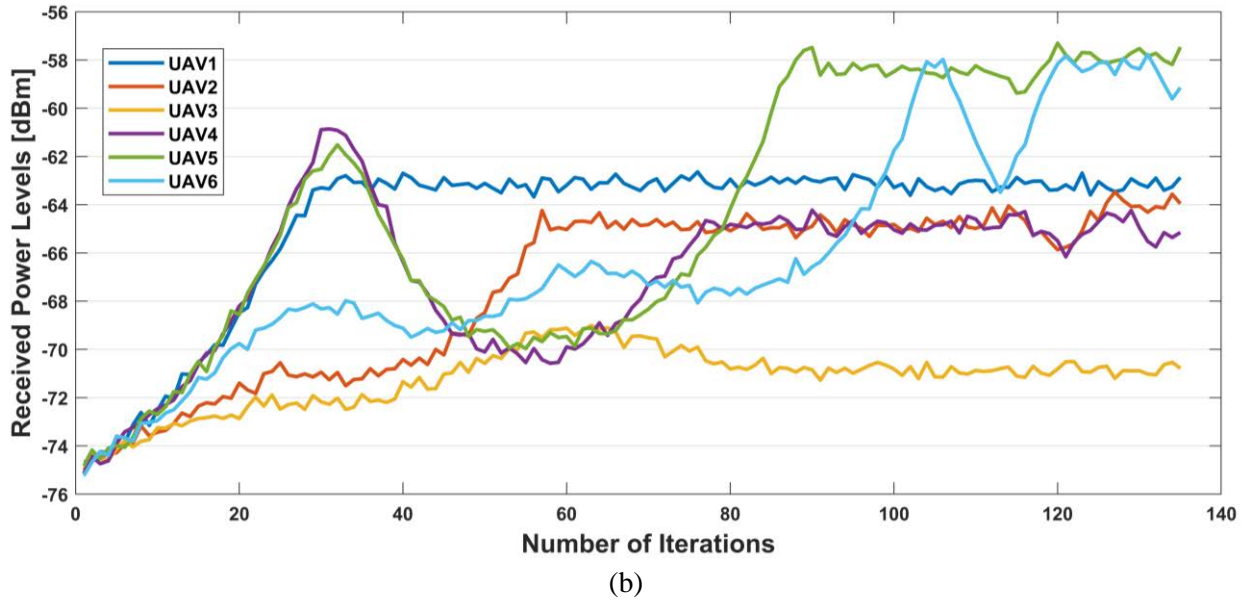
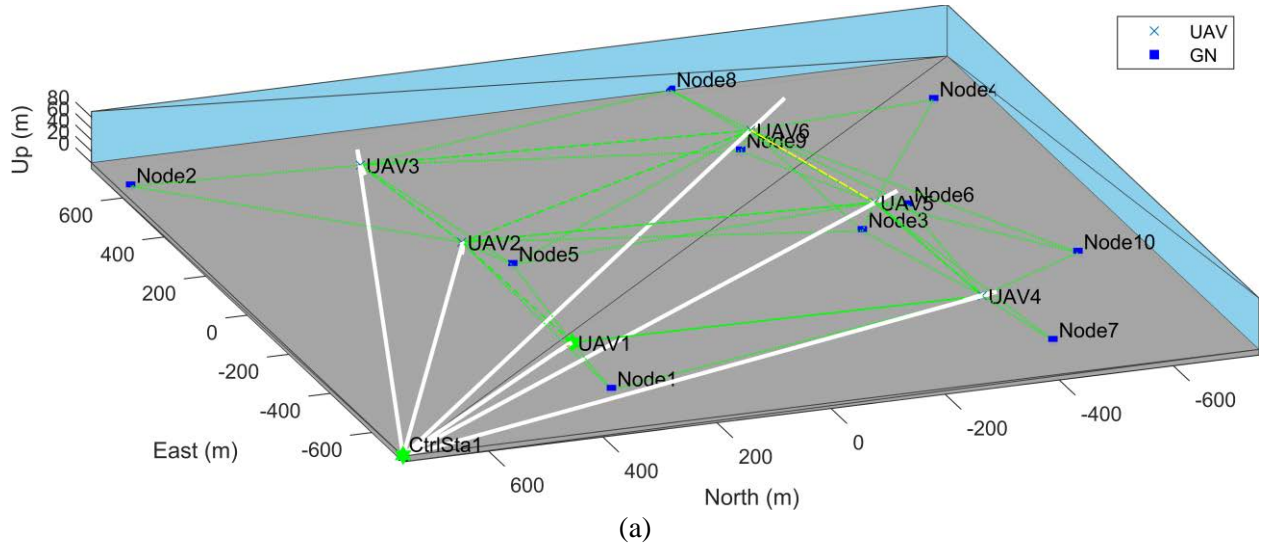


Figure 4.16: Topology and convergence results when using a log-normal propagation model (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

As shown in Figures 4.15(a) and 4.16(a), the network topology after convergence looks very similar for both alternatives. However, in a somehow counterintuitive manner, a faster convergence is achieved when using the log-normal propagation model (141 vs. 135 iterations), as can be seen in Figures 4.15(b) and 4.16(b). This improved convergence could be attributed to the log-normal model presenting more distinctive values of received power levels at each position, that lead to less conflicting objective function values.

### 4.2.3 Stopping Criteria Alternatives

As mentioned in Section 3.4.5, the PSO algorithm stops when any of the three following conditions is met: (i) the objective function has not improved for a certain time (the maximum stall count has been reached), (ii) when the maximum number of iterations has been reached, or (iii) when the relative change in fitness is below a certain threshold. Two alternative fitness functions were developed to quantify the relative change in fitness: (i) level of relative change in total received power in the last iteration with respect to the average over the previous ten iterations, as a measure of signal stability defined by Equation (3.11), and (ii) average distance of the current and previous ten positions to the center of gravity of the previous ten positions, as a measure of position stability as defined by Equation (3.12). Their performance is compared in this section using the log-normal propagation model, as it provides more challenging conditions for both of them.

Table 4.13: Stopping criteria alternatives using adaptive inertia weight and log-normal propagation.

| <b>Parameter</b>                               | <b>Signal Stability<br/>Equation (3.11)</b> | <b>Position Stability<br/>Equation (3.12)</b> |
|--|---|---|
| Grid size                                      | 1500 m x 1500 m                             |   |
| Mobility algorithm                             | Hybrid PSO                                  |   |
| Coverage gain                                  | Equation (3.7)                              |   |
| Path maintenance gain                          | Equation (3.7)                              |   |
| Excessive closeness penalty                    | Equation (3.8)                              |   |
| UAV <sub>1</sub> connection to Control Station | Equation (3.9)                              |   |
| Propagation Model                              | Log-normal                                  |   |
| Stopping criteria reached                      | Max. iterations                             | Convergence                                   |
| Number of iterations                           | 180   | 135   |

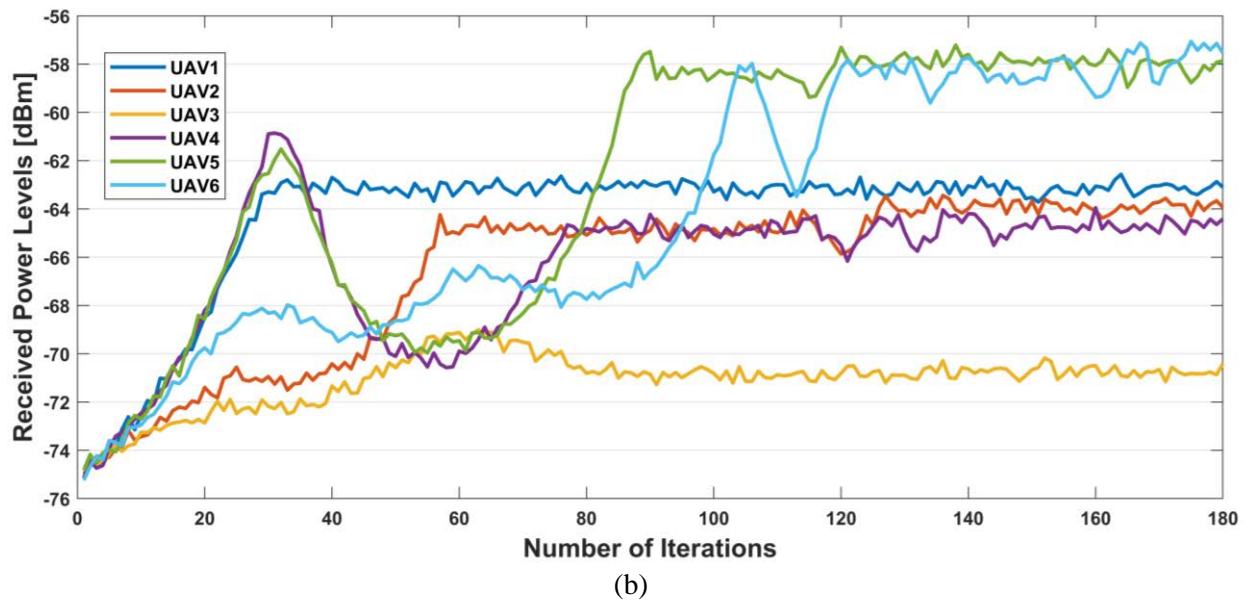
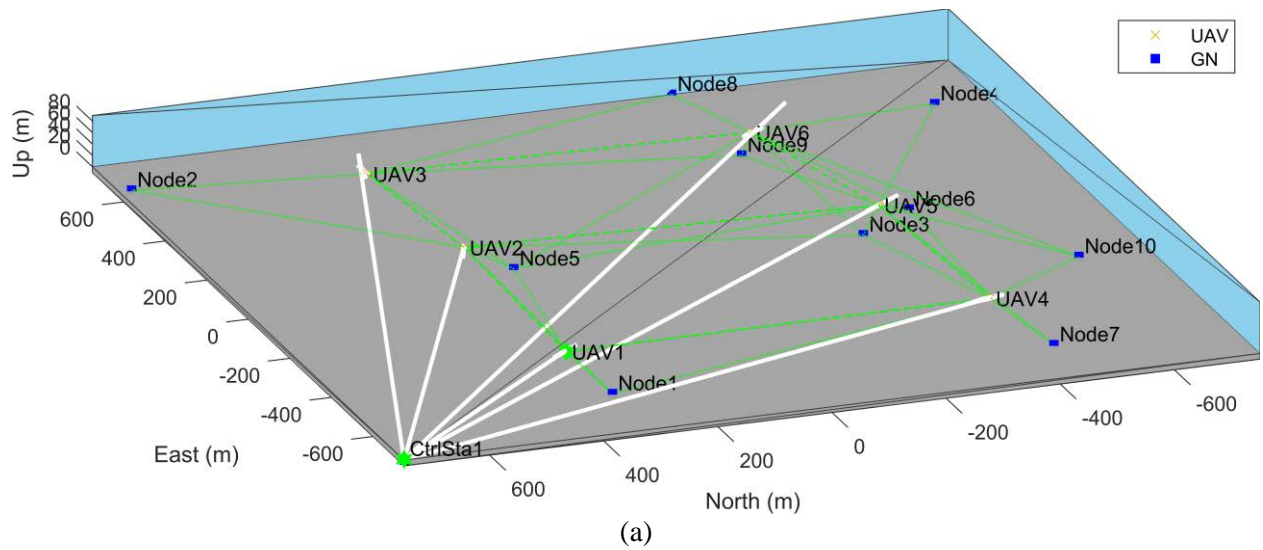


Figure 4.17: Topology and convergence results when using signal stability as fitness function (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

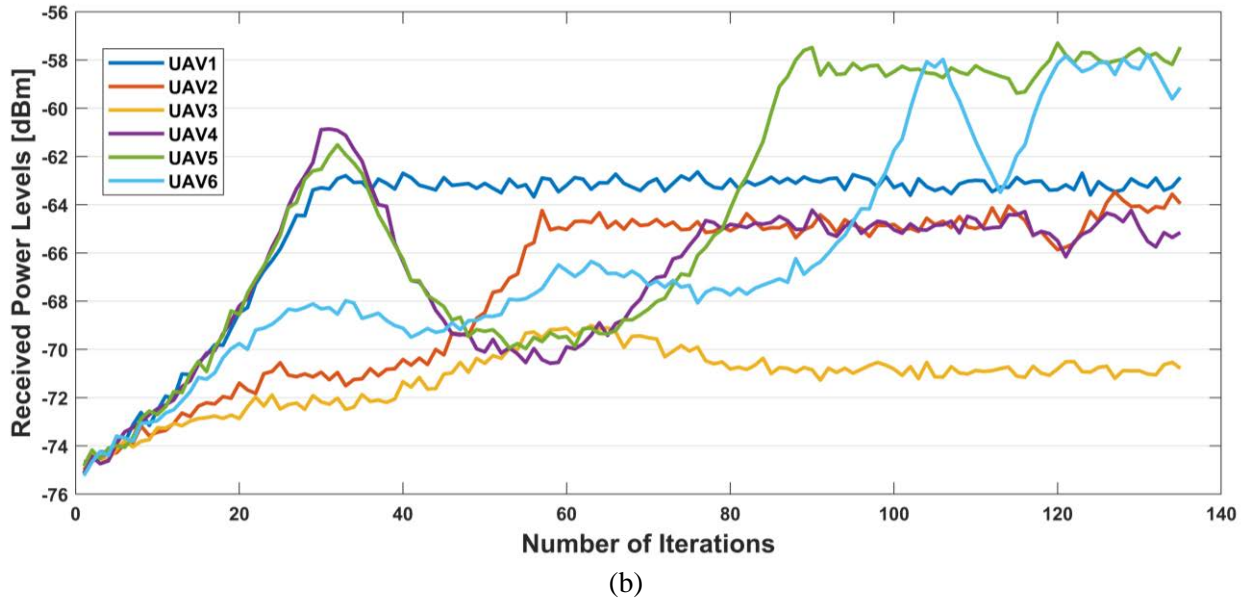
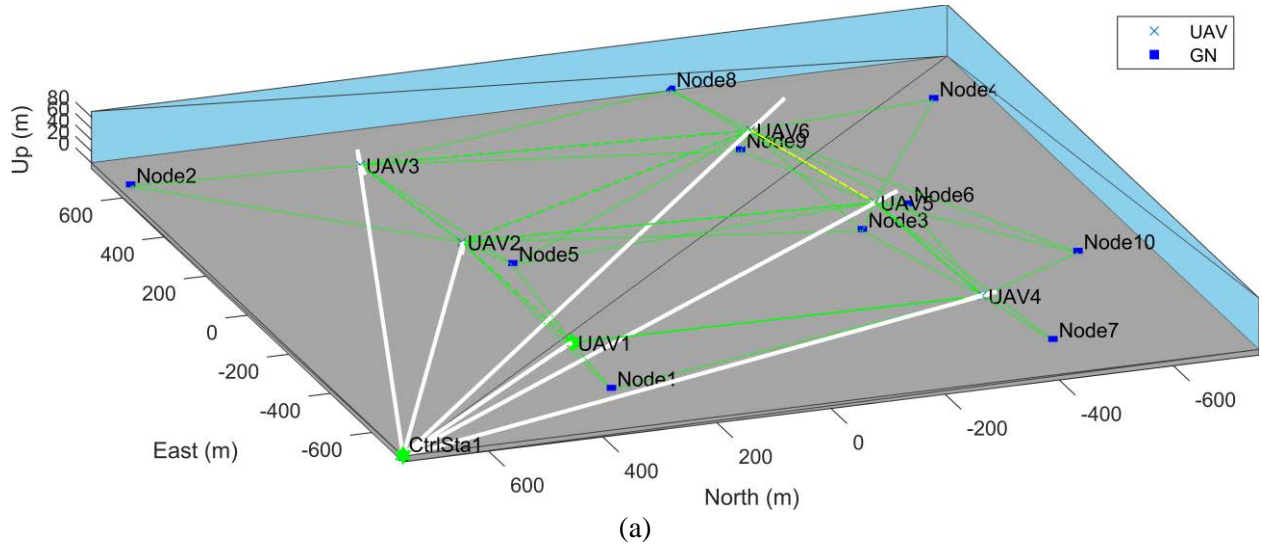


Figure 4.18: Topology and convergence results when using position stability as fitness function (Hybrid PSO). (a) Network topology after stopping criteria are met. (b) Aggregated received power levels at each UAV.

As shown in Figures 4.17(a) and 4.18(a), the network topology after convergence looks very similar for both alternatives. However, a faster convergence is achieved when using the average position change as stopping criterion (135 vs. 180 iterations), as can be seen in Figures 4.17(b) and 4.18(b). In fact, convergence is not reached when using the relative change in aggregated received power, due to the stochastic variations in the signal. These results suggest that the random component present in real-world radio signals can lead to signal stability not being achieved even

when position stability may have been attained. The threshold to consider a signal as stable could be increased to a certain amount, but that amount depends on changing propagation conditions. On the other hand, the fitness threshold to determine position stability depends only on the UAV maximum speed, which is a previously known and fixed value. Therefore, the average position change, as defined in Section 3.4.5 and Equation (3.12), will be used for the performance analysis.

### 4.3 Performance Metrics

Once the objective function has been defined and the PSO parameters have been configured, the mobility algorithm can be applied, and the effectiveness and efficiency of the solution can be assessed in terms of performance metrics. The performance of the proposed FANET is evaluated in terms of the objectives defined in the problem formulation: maximizing coverage and maintaining a communications path to the control station. Therefore, the following metrics have been defined:

#### 4.3.1 Number of Iterations Required for Stabilization or Stoppage ( $t_s$ )

This metric represents the amount of time or iterations required by the mobility optimization algorithm to find a solution or in other words, the time required to achieve position stability. It is determined by the stopping criteria and provides a measure of the FANETs' time efficiency.

#### 4.3.2 Percentage of Covered Ground Nodes ( $C\%$ )

This metric represents the FANET effectiveness in providing coverage to the ground nodes. It is defined as the percentage of ground nodes covered by at least one UAV after  $t_s$ , as shown in Equation (4.1).

$$C\% = \frac{n_f}{n} \times 100. \quad (4.1)$$

where  $n_f$  is the number of ground nodes covered by at least one UAV after  $t_s$ , and  $n$  is the total number of ground nodes.

### 4.3.3 FANET Coverage Efficiency ( $\eta_c$ )

For the purpose of this research, the FANET is held to be more efficient if a higher percentage of ground nodes are covered, without having ground nodes being covered by more than one UAV, and without having UAVs not covering any ground node. Under certain circumstances, a degree of redundancy and reserve may be desirable. However, for the case of maximizing coverage with the least number of UAVs, having ground nodes being covered by more than one UAV is considered redundant, and UAVs not covering at least one ground node are considered idle.

As shown in Equation (4.2), FANET coverage efficiency is defined as the ratio of ground nodes covered by at least one UAV after  $t_s$  to the total number of ground nodes (a measure of coverage effectiveness), times the ratio of covered ground nodes to the total number of UAV connections with ground nodes (a measure of redundancy), times the ratio of UAVs that have connections with ground nodes to the total number of UAVs (a measure of inactivity).

$$\eta_c = \frac{n_f}{n} \times \frac{n_f}{n_s} \times \frac{N_f}{N} \quad (4.2)$$

where  $n_f$  is the number of ground nodes covered by at least one UAV after  $t_s$ ,  $n$  is the total number of ground nodes,  $n_s$  is the total number of UAV connections with ground nodes,  $N_f$  is the number of UAVs that have connections with ground nodes, and  $N$  is the total number of UAVs.

The coverage efficiency value ranges from 0 to 1, with a higher value indicating a higher percentage of ground nodes being covered with less redundant links and with less idle UAVs.

#### 4.3.4 Percentage of Ground Nodes with a Path to the Control Station ( $P\%$ )

This metric represents the FANET effectiveness in providing the ground nodes with a path to the control station. It is defined as the percentage of ground nodes that have a path to the control station after  $t_s$ , as shown in Equation (4.3).

$$P\% = \frac{n_p}{n} \times 100. \quad (4.3)$$

where  $n_p$  is the number of ground nodes that have a path to the control station after  $t_s$ , and  $n$  is the total number of ground nodes.

#### 4.3.5 Overall Efficiency ( $\eta_o$ )

This metric represents the overall FANET efficiency regarding fulfilment of the coverage maximization and path maintenance objectives. It is defined as the coverage efficiency,  $\eta_c$ , times the ratio of ground nodes that have a path to the control station to the total number of ground nodes, as presented in Equation (4.4).

$$\eta_o = \eta_c \times \frac{n_p}{n} \quad (4.4)$$
$$\eta_o = \frac{n_f}{n} \times \frac{n_f}{n_s} \times \frac{N_f}{N} \times \frac{n_p}{n}$$

where  $\eta_c$  is the FANET Coverage Efficiency after  $t_s$ ,  $n_p$  is the number of ground nodes that have a path to the control station after  $t_s$ , and  $n$  is the total number of ground nodes.

The overall efficiency value ranges from 0 to 1, with a higher value indicating a higher percentage of ground nodes being covered with less redundant links, with less idle UAVs, and with more ground nodes having a path to the control station.

## 4.4 Performance Evaluation of the Different Mobility Algorithms

The best alternatives for objective function, PSO parameters configuration, stopping criteria, and propagation models have been determined and summarized in Table 4.14. Consequently, the performance is assessed under the three scenarios described in Table 4.15, according to the model parameters presented in Table 4.16, for the following three mobility algorithms:

- Fixed trajectory: The UAVs fly through a fixed trajectory.
- PSO-only: The UAVs fly applying PSO from the start.
- Hybrid PSO: The UAVs fly through a fixed trajectory first and apply PSO afterwards.

The fixed trajectory is generated using Algorithm 3.6 as defined in Section 3.4.4. Each of the three mobility algorithms is assessed under the three scenarios described in. The results are tabulated in APPENDIX F and analyzed in the following subsections of this chapter.

Table 4.14: Objective function and PSO configuration.

| Component          | Description                | Equation                |
|--------------------|----------------------------|-------------------------|
| Objective function | Distance-based gain        | (3.7), (3.8), and (3.9) |
| PSO configuration  | Adaptive inertia weight    | ---                     |
| Propagation models | Ray tracing and log-normal | (2.4)                   |
| Stopping criteria  | Position stability         | (3.12)                  |

Table 4.15: Assessment scenarios.

| Identification    | Propagation Model | Access Network | Backhaul Network | Backbone |
|-------------------|-------------------|----------------|------------------|----------|
| Ray tracing       | Ray tracing       | Wi-Fi          | Wi-Fi            | Wi-Fi    |
| Log-normal        | Log-normal        | Wi-Fi          | Wi-Fi            | Wi-Fi    |
| Log-normal + LoRa | Log-normal        | Wi-Fi          | LoRa             | Wi-Fi    |



Table 4.16: Final model configuration parameters.

| General Parameters           |   |                         |                   |
|------------------------------|---|-------------------------|-------------------|
| Grid size                    | 2000 m x 2000 m   | Overall max. height     | 121 m (400 ft.)   |
| Number of UAVs               | 1-25  | Initial speed           | 30 m/s            |
| Number of ground nodes       | 10, 20, 30  | Maximum speed           | 45 m/s            |
| Fixed trajectory max. height | 60 m  | Time step               | 1 s               |
| Radio Parameters             |   |                         |                   |
| Backhaul technology          |   | Wi-Fi                   | LoRa <sup>2</sup> |
| Frequency                    |   | 2.437 GHz               | 900 MHz           |
| Tx power                     |   | 15 dBm                  | 15 dBm            |
| Tx gain                      |   | 1 dB                    | 1 dB              |
| Receiver sensitivity         |   | -82 dBm                 | -105 dBm          |
| Rx gain                      |   | 0 dB                    | 0 dB              |
| Propagation model            |   | Ray tracing, log-normal |                   |
| PSO Parameters               |   |                         |                   |
| Inertial weight (w)          | 0.95  | Max. stall count        | 150               |
| Individual weight (c1)       | 1.35  | Max. iterations         | 180               |
| Group weight (c2)            | 0.01  |                         |                   |
| Stopping criteria            | Mean distance to CoG, max. stall count, max. number of iterations |                         |                   |

#### 4.4.1 Number of Iterations

The three mobility algorithms have been tested regarding the number of iterations required for stabilization or stoppage. Figure 4.19 shows that the number of iterations remains the same for the fixed trajectory flight regardless of the propagation model or the backhaul network employed, as it is only dependent on the size of the grid and the number of UAVs. The fixed trajectory algorithm, in general, performs better than the other two regarding the number of iterations. However, it is worth noting that the PSO-only algorithm converges faster for a single UAV, and that the number of iterations for the hybrid algorithm with a single UAV is roughly the sum of the number of iterations required for both the fixed-trajectory and the PSO-only algorithms. Also, when a larger number of UAVs is deployed, the maximum number of iterations is reached before position stabilization is achieved. This saturation occurs with both the hybrid and PSO-only algorithms,

<sup>2</sup> When applicable

and could be attributed to the increased rate at which UAVs dynamically change clusters leading to higher instability.

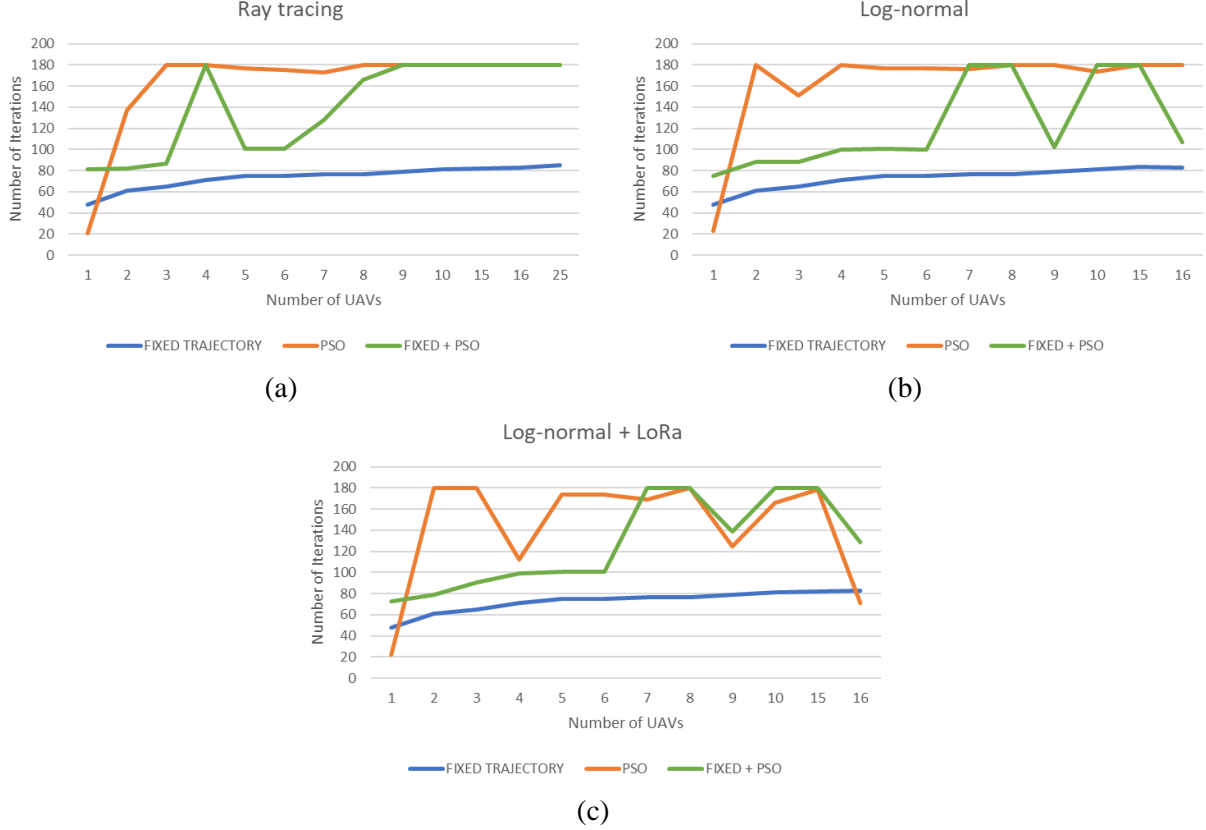


Figure 4.19: Number of iterations required for stabilization or stoppage ( $t_s$ ). (a) Ray tracing propagation model. (b) Log-normal propagation model. (c) Log-normal propagation model with LoRa backhaul.

Figures 4.19(b) and 4.19(c) show that there are exceptions to this grid saturation. In the case of 9 UAVs, which are arranged in a square-grid formation at the maximum height of the fixed trajectory, the proximity and connectivity between the UAVs are higher compared to other configurations. This allows for higher information exchange among UAVs, enabling them to converge more quickly towards an optimal solution. A similar scenario that enhances the cooperation among the swarm happens for 16 UAVs.

Another potential cause for saturation when a larger number of UAVs is deployed lies in the penalties for excessive closeness that might conflict with the path maintenance objective, although

the exceptions mentioned in the previous paragraph seem to rebuff this possibility. Nonetheless, an increase in the number of iterations up to a tolerable limit can be an acceptable byproduct of applying penalties for excessive closeness, if a suitable solution is found with regard to coverage and path maintenance, or if a better solution is found using less UAVs.

#### 4.4.2 Percentage of Covered Ground Nodes

The plots in Figures 4.20(b) and 4.20(c) show that the percentage of covered ground nodes remains identical for the fixed trajectory algorithm (blue line), as coverage depends on the UAVs' final positions, the propagation model, and the access network, which are the same in both cases.

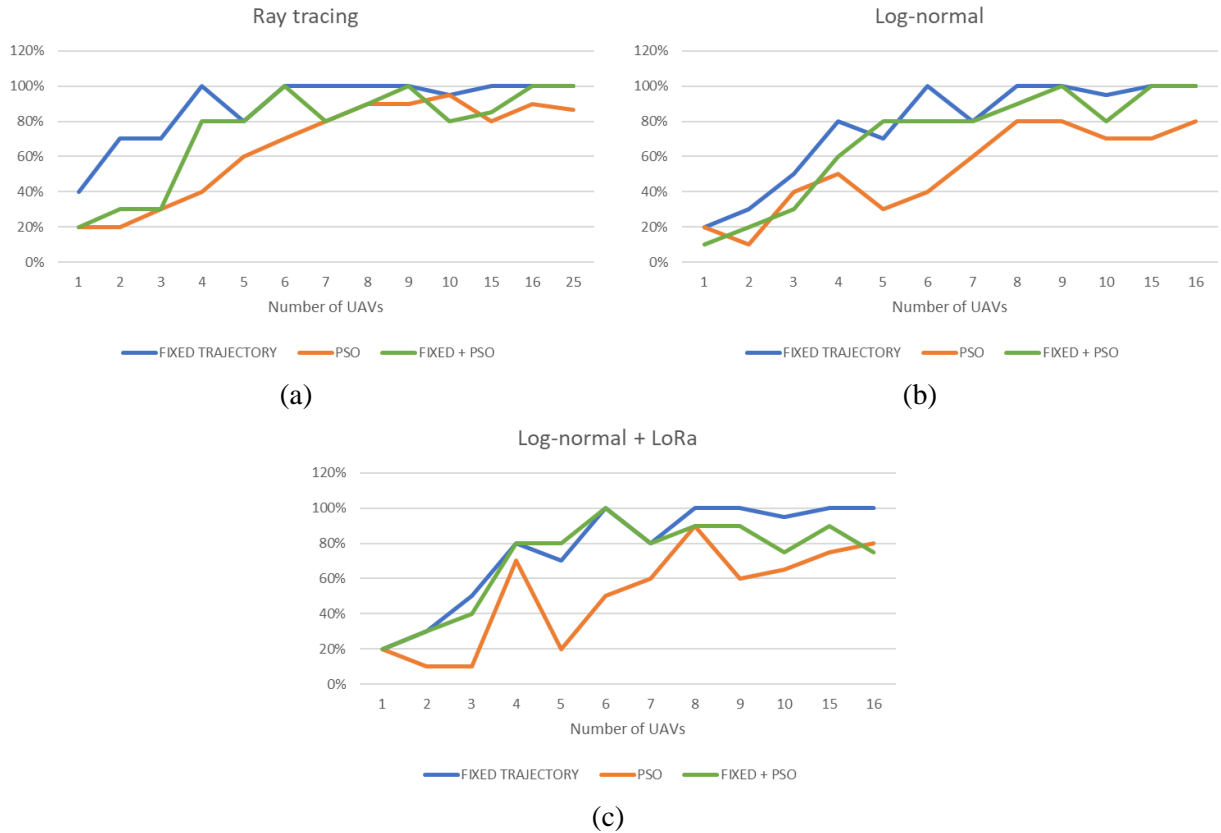


Figure 4.20: Percentage of covered ground nodes ( $C\%$ ). (a) Ray tracing propagation model. (b) Log-normal propagation model. (c) Log-normal propagation model with LoRa backhaul.

The fixed trajectory has shown to provide the best coverage results across all three scenarios. However, the Hybrid PSO performance becomes comparable to that of the fixed algorithm, with

the hybrid algorithm outperforming the fixed trajectory flight for a swarm made up of 5 UAVs. When a 5-UAV swarm is employed, the arrangement at the maximum height of the fixed trajectory is asymmetrical and one grid segment is left uncovered by the fixed trajectory algorithm. The fact that the Hybrid PSO outperforms the fixed trajectory algorithm for this configuration indicates that the adaptive nature of the Hybrid PSO allows it to perform better under irregular conditions.

#### **4.4.3 FANET Coverage Efficiency**

From the results shown in Figure 4.21, it is reasonable to expect a higher coverage efficiency with the fixed trajectory algorithm, as the likelihood of having more than one UAV covering the same ground node is lower when the UAVs are uniformly distributed above the grid—even though deterministically—and the ground nodes are also uniformly distributed—even though randomly. Although it has not been tested, it is expected that PSO-only and Hybrid PSO algorithms outperform the fixed trajectory algorithm for random distributions of ground nodes other than uniform.

Additionally, it is observed that the coverage efficiency decreases with the increase in the number of UAVs, irrespective of the propagation model. This is attributed to a higher probability of UAVs having overlapping coverage areas. Multiple UAVs may end up covering the same ground nodes, resulting in inefficient resource utilization. This redundancy leads to diminishing returns in terms of coverage efficiency. Furthermore, in a real-world implementation, as the number of UAVs increases, the likelihood of interference between their communication links would also increase, leading to a disruption in overall communication and thus affecting coverage efficiency and convergence time.

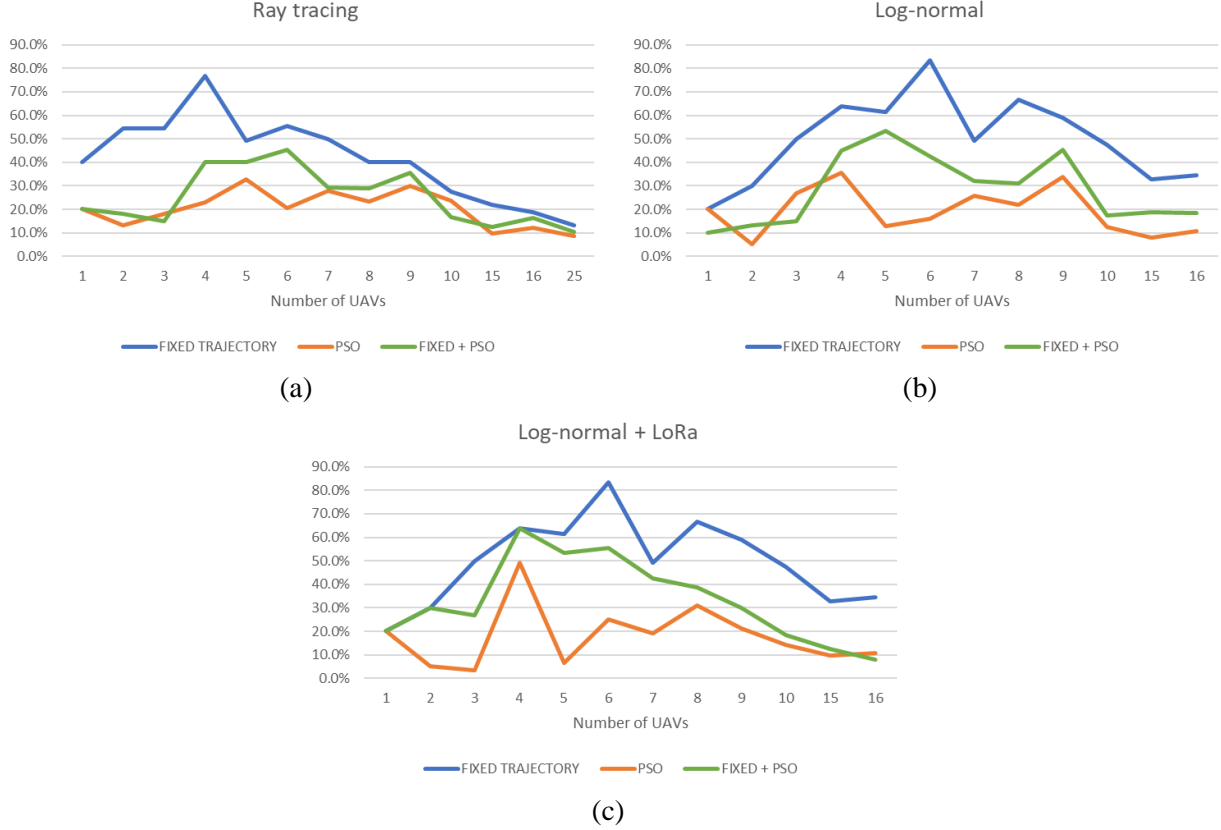


Figure 4.21: FANET Coverage Efficiency ( $\eta_c$ ). (a) Ray tracing propagation model. (b) Log-normal propagation model. (c) Log-normal propagation model with LoRa backhaul.

#### 4.4.4 Percentage of Ground Nodes with a Path to the Control Station

The fixed trajectory algorithm achieves coverage maximization by uniformly distributing the UAVs above the grid. Nonetheless, it does not take path maintenance into consideration, and only attains it circumstantially when the number of UAVs is sufficiently large as to allow the establishment of links between adjacent UAVs as well as between the control station and at least one UAV. Hence, the Hybrid PSO and the PSO-only algorithms generally outperform the fixed trajectory flight in the percentage of ground nodes with a path to the control station, especially under a more conservative propagation model intended to represent less ideal propagation conditions. However, when a larger number of UAVs is deployed, the increased instability results in a performance decay of the PSO-based algorithms for the ray tracing propagation model and for

the log-normal propagation model when using LoRa as a Backhaul, as shown in Figures 4.22(a) and 4.22(b).

Under the assumption of a flat earth model without any obstacles, the ray tracing propagation model produces path loss values similar to those of the free-space model. The decrease in path loss values results in enhanced air-to-air connectivity among neighboring UAVs when employing the fixed trajectory algorithm. Consequently, as shown in Figure 4.22(a), a path to the control station is established for most ground nodes when the number of UAVs is large enough. Nevertheless, when using a more conservative propagation model, the Hybrid PSO clearly outperforms the fixed trajectory flight as shown in Figure 4.22(b).

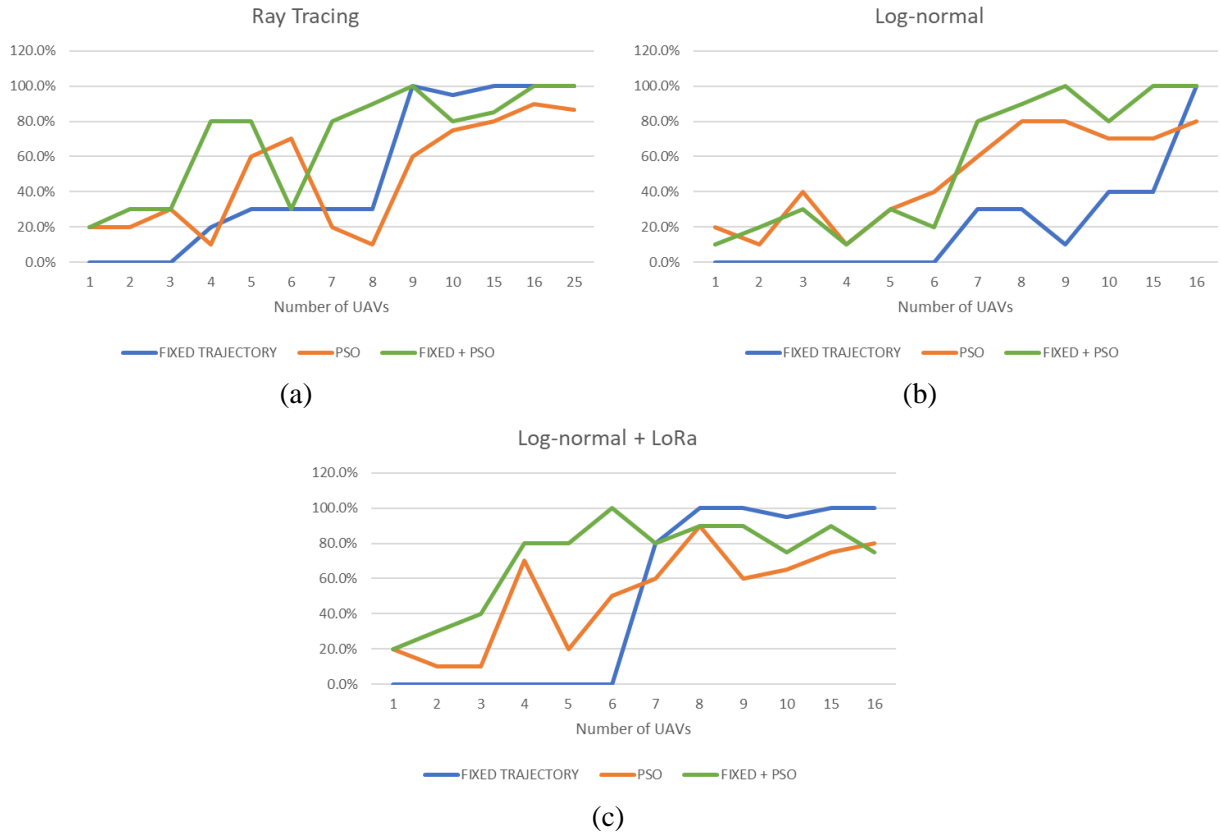


Figure 4.22: Percentage of ground nodes with a path to the control station ( $P\%$ ). (a) Ray tracing propagation model. (b) Log-normal propagation model. (c) Log-normal propagation model with LoRa backhaul.

When LoRa is used as backhaul, the air-to-air link distance is increased, which is equivalent to reducing the size of the grid. Under these circumstances, the performance of the three algorithms for a larger number of UAVs is affected consequently, with the fixed trajectory flight improving and the PSO-based algorithms deteriorating. Still, the use of a LoRa backhaul can be worthy of a compromise, if an acceptable solution is found with regard to coverage and overall efficiency, or if a comparable solution is found using less UAVs.

#### **4.4.5 Overall Efficiency**

The overall efficiency is a combination of coverage efficiency,  $\eta_c$ , and the percentage of ground nodes with a path to the control station,  $P\%$ . Thus, the results can be interpreted similarly to the previous sections, with the Hybrid PSO algorithm clearly outperforming the other two, particularly under the log-normal propagation model, that represents less favorable propagation conditions despite the ideal conditions of the model scenario.

The Hybrid PSO outperforms the others in all three assessment scenarios as long as the number of UAVs does not exceed 7–9. When the number of UAVs increases beyond 7–9 for the given grid size, all algorithms' efficiency decreases, but the PSO-based algorithms' performance is affected the most by augmented instability. Thus, it is important to identify the adequate number of UAVs, given the grid size and propagation conditions.

The use of LoRa in air-to-air links increases the communication range between UAVs, equivalently reducing the grid size, which also increases instability for the PSO-based algorithms when using a larger number of drones. Therefore, when using LoRa, the number of UAVs can be reduced or the grid size increased, compared to the number of UAVs or grid size when not using such technology.

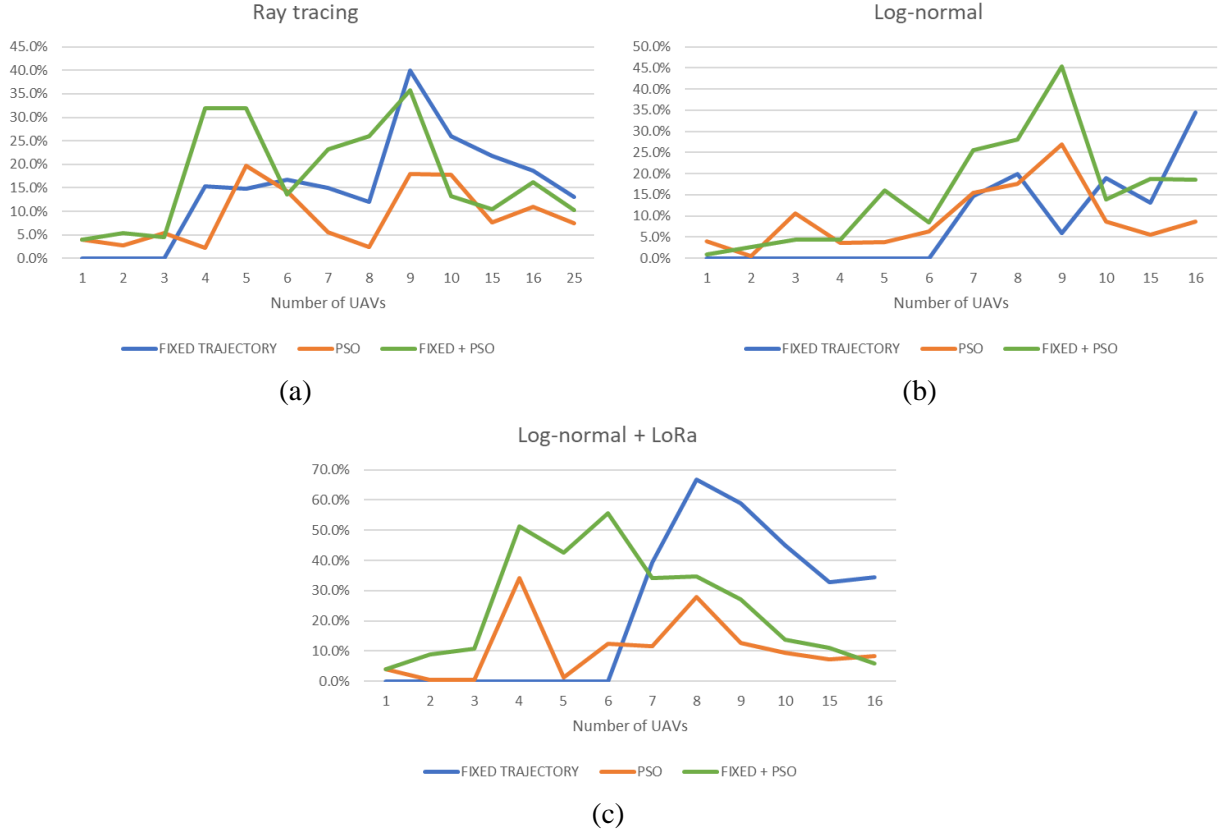


Figure 4.23: Overall efficiency ( $\eta_0$ ). (a) Ray tracing propagation model. (b) Log-normal propagation model. (c) Log-normal propagation model with LoRa backhaul.

To achieve overall efficiency maximization inside a 2 km by 2 km area, 9 UAVs could be deployed on an open rural scenario, represented by the ray tracing propagation model, using either the Hybrid PSO or the fixed trajectory algorithms. In a suburban scenario, represented by the log-normal propagation model, 9 UAVs perform better in the same area using the Hybrid PSO. In a suburban scenario, represented by the log-normal propagation model, where applications such as short messaging or location services can tolerate lower data rates, 6 UAVs could be deployed using LoRa technology as backhaul and the Hybrid PSO algorithm for mobility, or 8 UAVs using the fixed trajectory algorithm if resources are available.



# **Chapter 5**

## **Conclusions and Future Work**

The mobility of UAVs in FANETs is crucial for network performance, and optimizing the mobility patterns of UAVs can significantly enhance the network's efficiency. PSO is a bio-inspired optimization algorithm that has gained popularity in solving various optimization problems. A comprehensive simulation environment was developed in MATLAB where three mobility algorithms have been compared based on ground coverage and topology formation. With 9 UAVs and 10 ground nodes in a 2000 m x 2000 m grid, the Hybrid PSO achieves a 45% overall network efficiency that is reduced only by redundant air-to-ground links. Furthermore, with this configuration all ground nodes are covered and all of them have a path to the control station in approximately 100 iterations, which are equivalent to less than 2 minutes under the simulation conditions, and only around 20 seconds higher than the time it would take for the fixed trajectory flight to reach their final positions. Therefore, the Hybrid PSO is a promising alternative to achieve FANET mobility while maximizing coverage to fixed ground nodes and creating a path to a control station. Further conclusions and future work by subject are presented in the following subsections.

### **5.1 Conclusions**

#### **5.1.1 Regarding the Objective Function and Multiobjective Optimization**

Defining the objective function is a crucial step in optimization as it lays the foundation for the entire process. However, setting parameters for the optimization algorithm is also a significant step and is often connected to defining the objective function. These processes require careful consideration and testing as many parameters come into play, and their impact must be systematically analyzed one at a time. The task of setting parameters and defining the objective

function can be challenging but is essential for achieving successful optimization results. Therefore, it is important to take the time to carefully define the objective function and set the appropriate parameters for the optimization algorithm to ensure optimal results.

Multiobjective optimization involves solving problems that have multiple objectives that cannot be optimized simultaneously. These objectives often conflict with one another, and finding a solution that balances them is challenging. To accurately solve such problems, the objective function should represent the trade-offs between the different objectives accurately. In other words, the objective function should provide a way to measure how much progress is made towards one objective while sacrificing some progress towards the others. By properly representing the trade-offs, it becomes possible to identify and analyze the most effective solutions that achieve the optimal balance between all objectives. Therefore, in multiobjective optimization, it is crucial to ensure that the objective function accurately represents the trade-offs between the different objectives.

The use of a single objective function to achieve multiobjective optimization has its advantages, as it is a straightforward concept that provides a clear direction for optimization. However, this approach has some significant disadvantages that must be considered. For example, it can be challenging to tune the objective function properly to ensure that it provides optimal results. Additionally, there is no clear reference for the range of values that the objective function can take. As a result, it cannot be used to define an absolute fitness level, but only a relative one. These disadvantages highlight the importance of carefully considering the objective function's design and implementation to ensure that it provides the necessary information to guide optimization effectively. Therefore, while the concept of using an objective function is straightforward, it requires careful consideration to ensure that it is effective in practice.

Applying the expected distance-based quadratic gain when a path to the control station exists produces better results in convergence time for different grid sizes than applying it just when more than one node is found, particularly for the PSO-only algorithm. However, keeping a base-2 exponential gain when more than one node is found is also necessary to improve the balance between objectives.

Applying a range-based penalty to all UAVs' for excessive closeness helps to maintain the UAVs within a desirable range from each other, which in turn results in a more efficient topology by reducing redundant coverage. This does not seem to guarantee collision avoidance, though, given that the solution algorithm does not replace previous personal or group best values that draw a UAV to the positions where those values were previously recorded.

Applying the expected distance-based quadratic gain to UAV<sub>1</sub>'s function when it is in the optimal range from the control station improves the likelihood of having a path to it from the covered ground nodes, and also improves convergence time.

### **5.1.2 Regarding the Optimization Algorithm**

Including constraints in the optimization algorithm allows for the incorporation of real-world limitations and practical considerations into the optimization problem. Reducing speed to a maximum physical constraint is not mandatory in a real-world FANET implementation, as the speed would be limited naturally. However, a stall counter and an adaptive inertia weight are useful in applications with a reduced number of particles over large search spaces.

Setting up parameters in the optimization algorithm is critical for obtaining good results in the optimization process. The range of values defined for  $w$ ,  $c_1$  and  $c_2$  in Equation (3.13) guarantee convergence, but do not necessarily shorten convergence time, particularly for large search spaces.

During initial tests, a value of -0.05 was set for  $c_2$  to cause a repelling effect between UAVs. However, a small attraction value of 0.01 was ultimately configured that resulted in improved path formation.

The inertia weight,  $w$ , must also be limited to keep it from growing excessively, which might lead to low spatial resolution and increased instability, even when the maximum speed is constrained. Constraining speed below its maximum value might be a requirement when energy efficiency is part of the problem objectives.

A higher temporal resolution improves the Hybrid PSO performance in the simulation. It takes longer for the simulation to run, but collecting more and possibly better readings might produce enhanced results. Figure 4.8(a) shows that all ground nodes have a path to the control station after convergence using 6 UAVs for the Hybrid PSO algorithm. A time step of 0.5 seconds was established in that simulation in order to test the best alternative for the objective function. However, a lower temporal resolution with a time step of 1 second was set for the performance assessment of the different mobility algorithms, in order to reduce the simulation time. As a result, only 20 percent of the nodes have a path to the control station after convergence for the same number of UAVs, grid size, and propagation model, as shown in Figure 4.22(a). Increasing the maximum speed beyond practical values can produce a similar consequence, as it lowers spatial resolution, even if the temporal resolution is kept at the same value.

Regarding the stopping criteria, using the mean distance to the center of gravity of the current and previous positions as a stopping criterion results in shortened convergence time. Also, it performs better when using stochastic propagation models, such as log-normal, that incorporate randomness in the received signal. This suggests that it would also perform better in a practical implementation.

### **5.1.3 Regarding Overall Performance**

Regarding coverage efficiency, although it has not been tested, it is expected that the PSO-only and Hybrid PSO algorithms will outperform the fixed trajectory algorithm for node distributions other than randomly uniform.

The fixed trajectory algorithm outperforms the Hybrid-PSO and PSO-only in time and coverage metrics but does not achieve multiobjective optimization purposefully, and only attains it circumstantially when the number of UAVs is sufficiently large as to allow the establishment of links between adjacent UAVs as well as between the control station and at least one UAV. Nevertheless, it is a good alternative when resources are not a constraint.

However, when path formation to the control station is taken into consideration, the Hybrid-PSO outperforms the others, except when LoRa comes into place. LoRa increases horizontal range, which is comparable to making the grid smaller. As a result, the fixed trajectory outperforms the others again. The Hybrid PSO does not perform well when the number of UAVs increases excessively with respect to the grid size.

### **5.1.4 Regarding the Use of LoRa**

When LoRa is used as backhaul, the air-to-air link distance is increased, which is equivalent to reducing the size of the grid, which affects the performance of the PSO-based algorithms for a larger number of UAVs. Still, the use of a LoRa backhaul can be worthy of a compromise, if an acceptable solution is found with regard to coverage and overall efficiency, or if a comparable solution is found using less UAVs.

LoRa technology offers significant benefits within the FANET context in maximizing the communications range between UAVs, particularly in low-data-rate applications like WSN, remote control, flight coordination, and drone identification. Its long-range capabilities make it

well-suited for these specific use cases. However, when considering the use of LoRa as a FANET access network, it's important to acknowledge that its adoption among end users is not as widespread compared to other types of communication technologies. As a result, its applicability in certain domains may be limited.

In scenarios where human-oriented communications are crucial, a hybrid network that combines different communication technologies could be more convenient. For example, Wi-Fi or cellular networks, which are widely used and accessible to a large number of users, can be integrated alongside LoRa. This hybrid approach would be beneficial in applications such as localization, short messaging, or search and rescue operations, where the access network needs to cater to widespread usage and enable efficient communication between UAVs and humans.

## **5.2 Future Work**

### **5.2.1 Communications**

In terms of communications, several areas can be explored further. One aspect is to include terrain and building maps in order to test different propagation models. Additionally, developing a system level simulation for technologies like LoRa, other LPWANs, LTE, or 5G could provide valuable insights.

Continuing the analysis of small-scale propagation effects for the FANET channel is another important avenue for future work. This can involve testing various parameters such as end-to-end throughput, packet delivery ratio (PDR), and latency. Additionally, exploring different levels of physical and MAC layer abstraction, improving routing for load balancing, and testing link level parameters subject to UAV movement would contribute to a comprehensive understanding of the system.

### **5.2.2 Mobility based on Multiobjective Optimization**

Additional research can be performed to further assess the performance of the multiobjective optimization approach to FANET mobility. It would be beneficial to test the performance of the proposed objective function for different grid sizes and numbers of ground nodes, in order to provide a measure of the system scalability.

Exploring different approaches to solve the multiobjective optimization problem, such as dynamic UAV team assignment with one team per objective, can provide insights into efficiency. It would also be valuable to include communications performance parameters like throughput, PDR, and latency in the objective function and test their inclusion as fitness measures in the stopping criteria. Energy optimization can also be included as part of the objective function, and testing the use of penalties only for the objective function is another avenue to explore. Automating the process of obtaining performance metrics would streamline the evaluation process.

Testing UAV dynamic models under various environmental conditions, such as windspeed, would enhance the understanding of their performance. Including features such as lidar, radar, and position-based collision avoidance can contribute to safer and more efficient mobility. Additionally, exploring the use of machine learning approaches, such as Support Vector Machine (SVM), to improve mobility efficiency would be worthwhile.

Considering that in many of the FANET applications described in Chapter 2, it is likely that the UAVs might be deployed from a single position, UAVs are initially located at one corner of the grid for this study. Nevertheless, the performance of deploying each UAV from a different initial position could be tested in further research.

Testing different heights for the fixed trajectory and determining the optimal height based on frequency, grid size, and the number of UAVs would further optimize the system. Incorporating maximum acceleration and deceleration constraints can also be considered.

### **5.2.3 Energy**

The energy aspect of FANETs is another area for future work. It would be beneficial to include low-battery UAV recharge and replacement mechanisms in the system. Testing energy replenishment methods like energy harvesting and solar power can contribute to sustainable and prolonged UAV operation.

### **5.2.4 The Use of LoRa in FANETs**

Exploring the use of LoRa in FANETs is another important focus. Testing the maximum communication ranges at the maximum data rate available for the latest LoRa chips in the 2.4 GHz band can provide insights into their performance. Additionally, investigating the concept of chirp modulation to improve the balance between data rate performance and energy consumption is valuable, as FANETs do not impose the same energy constraints as small battery-powered IoT sensors. One approach could involve reducing chirp duration by half.

As a final insight into future work, to carry out an experimental implementation would be instrumental to validate the results of this simulation-based study under practical conditions, as well as to tune the PSO and other configuration parameters such as time step, maximum UAV speed, fixed trajectory maximum height, and receiver sensitivity.



## REFERENCES

- [1] Rappaport, T.S. The wireless revolution. *IEEE Communications Magazine* **1991**, 29, 52-71, DOI 10.1109/35.109666. Available online: <https://ieeexplore.ieee.org/document/109666>.
- [2] George, T.D. Enabling the wireless revolution, 1993 International Symposium on VLSI Technology, Systems, and Applications, , Taipei, Taiwan, 12-14 May 1993; , pp. 7.
- [3] Gazis, V.; Görtz, M.; Huber, M.; Leonardi, A.; Mathioudakis, K.; Wiesmaier, A.; Zeiger, F.; Vasilomanolakis, E. A survey of technologies for the internet of things, 2015 International Wireless Communications and Mobile Computing Conference (IWCMC), 2015; , pp. 1090-1095.
- [4] Firouzi, F.; Farahani, B.; Weinberger, M.; DePace, G.; Aliee, F.S. IoT Fundamentals: Definitions, Architectures, Challenges, and Promises. In *Intelligent Internet of Things: From Device to Fog and Cloud*; Firouzi, F.; Chakrabarty, K.; Nassif, S., Eds.; Springer International Publishing: Cham, 2020; pp. 3-50.
- [5] Ansere, J.A.; Han, G.; Liu, L.; Peng, Y.; Kamal, M. Optimal Resource Allocation in Energy-Efficient Internet-of-Things Networks With Imperfect CSI. *IEEE Internet of Things Journal* **2020**, 7, 5401-5411.
- [6] Johnson, D.B. Routing in Ad Hoc Networks of Mobile Hosts, 1994 First Workshop on Mobile Computing Systems and Applications, , Santa Cruz, CA, USA, 8 - 9 December 1994; , pp. 158-163.
- [7] Tavli, B.; Heinzelman, W. Introduction. In *Mobile Ad Hoc Networks*; Tavli, B.; Heinzelman, W., Eds.; Springer: Dordrecht, The Netherlands, 2006; pp. 1-8.
- [8] Čolaković, A.; Hadžialić, M. Internet of Things (IoT): A review of enabling technologies, challenges, and open research issues. *Computer Networks* **2018**, 144, 17-39, DOI <https://doi.org/10.1016/j.comnet.2018.07.017>. Available online: <https://www.sciencedirect.com/science/article/pii/S1389128618305243>.
- [9] Fraccaroli, E.; Quaglia, D. Engineering IoT Networks. In *Intelligent Internet of Things: From Device to Fog and Cloud*; Firouzi, F.; Chakrabarty, K.; Nassif, S., Eds.; Springer International Publishing: Cham, 2020; pp. 97-171.
- [10] Adelantado, F.; Vilajosana, X.; Tuset-Peiro, P.; Martinez, B.; Melia-Segui, J.; Watteyne, T. Understanding the Limits of LoRaWAN. *IEEE Communications Magazine* **2017**, 55, 34-40, DOI 10.1109/MCOM.2017.1600613. Available online: <https://ieeexplore.ieee.org/document/8030482>.
- [11] Aggarwal, S.; Nasipuri, A. Survey and Performance Study of Emerging LPWAN Technologies for IoT Applications, 2019 IEEE 16th International Conference on Smart Cities: Improving Quality of Life Using ICT & IoT and AI (HONET-ICT), 2019; , pp. 69.

- [12] Ghazali, M.H.M.; Teoh, K.; Rahiman, W. A Systematic Review of Real-Time Deployments of UAV-Based LoRa Communication Network. *IEEE Access* **2021**, *9*, 124817-124830, DOI 10.1109/ACCESS.2021.3110872. Available online: <https://ieeexplore.ieee.org/document/9530545>.
- [13] Sundaram, J.P.S.; Du, W.; Zhao, Z. A Survey on LoRa Networking: Research Problems, Current Solutions, and Open Issues. *IEEE Communications Surveys & Tutorials* **2020**, *22*, 371-388, DOI 10.1109/COMST.2019.2949598. Available online: <https://ieeexplore.ieee.org/document/8883217>.
- [14] Centelles, R.P.; Freitag, F.; Meseguer, R.; Navarro, L. Beyond the Star of Stars: An Introduction to Multihop and Mesh for LoRa and LoRaWAN. *IEEE Pervasive Computing* **2021**, *20*, 63-72, DOI 10.1109/MPRV.2021.3063443. Available online: <https://ieeexplore.ieee.org/document/9385408>.
- [15] Tran, H.P.; Jung, W.; Yoo, D.; Oh, H. Design and Implementation of a Multi-Hop Real-Time LoRa Protocol for Dynamic LoRa Networks. *Sensors* **2022**, *22*, DOI 10.3390/s22093518.
- [16] Berto, R.; Napoletano, P.; Savi, M. A LoRa-Based Mesh Network for Peer-to-Peer Long-Range Communication. *Sensors* **2021**, *21*, DOI 10.3390/s21134314.
- [17] Liu, Y.; Liu, L.; Liang, J.; Chai, J.; Lei, X.; Zhang, H. High-Performance Long Range-Based Medium Access Control Layer Protocol. *Electronics* **2020**, *9*, DOI 10.3390/electronics9081273.
- [18] Macaraeg, K.C.V.G.; Hilario, C.A.G.; Ambatali, C.D.C. LoRa-based Mesh Network for Off-grid Emergency Communications, 2020 IEEE Global Humanitarian Technology Conference (GHTC), 2020; , pp. 1-4.
- [19] de Farias Medeiros, D.; Villarim, M.R.; de Carvalho, F.B.S.; de Souza, C.P. Implementation and Analysis of Routing Protocols for LoRa Wireless Mesh Networks, 2020 11th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2020; , pp. 20.
- [20] Chache, F.M.; Maxon, S.; Narayanan, R.M.; Bharadwaj, R. QoS Extension to a B.A.T.M.A.N. based LoRa Mesh Network, MILCOM 2021 - 2021 IEEE Military Communications Conference (MILCOM), 2021; , pp. 43-48.
- [21] Almeida, N.C.; Rolle, R.P.; Godoy, E.P.; Ferrari, P.; Sisinni, E. Proposal of a Hybrid LoRa Mesh / LoRaWAN Network, 2020 IEEE International Workshop on Metrology for Industry 4.0 & IoT, 2020; , pp. 702-707.
- [22] Pham, V.D.; Kisel, V.; Kirichek, R.; Koucheryavy, A.; Shestakov, A. Evaluation of A Mesh Network based on LoRa Technology, 2022 24th International Conference on Advanced Communication Technology (ICACT), 2022; , pp. 1280-1285.

- [23] Lundell, D.; Hedberg, A.; Nyberg, C.; Fitzgerald, E. A Routing Protocol for LoRA Mesh Networks, 2018 IEEE 19th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), 2018; , pp. 14-19.
- [24] Paredes, W.D.; Kaushal, H.; Vakulinia, I.; Prodanoff, Z. LoRa Technology in Flying Ad Hoc Networks: A Survey of Challenges and Open Issues. *Sensors* **2023**, *23*, 2403.
- [25] Chriki, A.; Touati, H.; Snoussi, H.; Kamoun, F. FANET: Communication, mobility models and security issues. *Computer Networks* **2019**, *163*, 106877, DOI <https://doi.org/10.1016/j.comnet.2019.106877>. Available online: <https://www.sciencedirect.com/science/article/pii/S1389128618309034>.
- [26] Guillen-Perez, A.; Cano, M. Flying Ad Hoc Networks: A New Domain for Network Communications. *Sensors* **2018**, *18*, DOI 10.3390/s18103571.
- [27] Srivastava, A.; Prakash, J. Future FANET with application and enabling techniques: Anatomization and sustainability issues. *Computer Science Review* **2021**, *39*, 100359, DOI <https://doi.org/10.1016/j.cosrev.2020.100359>. Available online: <https://www.sciencedirect.com/science/article/pii/S1574013720304597>.
- [28] Bekmezci, İ; Sahingoz, O.K.; Temel, Ş Flying Ad-Hoc Networks (FANETs): A survey. *Ad Hoc Networks* **2013**, *11*, 1254-1270, DOI <https://doi.org/10.1016/j.adhoc.2012.12.004>. Available online: <https://www.sciencedirect.com/science/article/pii/S1570870512002193>.
- [29] Wang, J.; Jiang, C. Introduction of Flying Ad Hoc Networks. In *Flying Ad Hoc Networks*; Shen, X.S., Ed.; Springer: Singapore, 2022; .
- [30] Khan, M.A.; Safi, A.; Qureshi, I.M.; Khan, I.U. Flying ad-hoc networks (FANETs): A review of communication architectures, and routing protocols, 2017 First International Conference on Latest trends in Electrical Engineering and Computing Technologies (INTELLECT), 2017; , pp. 1-9.
- [31] Khan, M.A.; Qureshi, I.M.; Khanzada, F. A Hybrid Communication Scheme for Efficient and Low-Cost Deployment of Future Flying Ad-Hoc Network (FANET). *Drones* **2019**, *3*, DOI 10.3390/drones3010016.
- [32] Al-Emadi, S.; Al-Mohannadi, A. Towards Enhancement of Network Communication Architectures and Routing Protocols for FANETs: A Survey, 2020; , pp. 1-10.
- [33] Wu, Q.; Zhang, M.; Dong, C.; Feng, Y.; Yuan, Y.; Feng, S.; Quek, T.Q.S. Routing protocol for heterogeneous FANETs with mobility prediction. *ChinaComm* **2022**, *19*, 186-201, DOI 10.23919/JCC.2022.01.014. Available online: <https://ieeexplore.ieee.org/document/9693479>.
- [34] Rappaport, T.S. *Wireless Communications-: Principles and Practice*, 2nd Edition ed.; Prentice-Hall: Upper Saddle River, NJ, USA, 2002; pp. 86.
- [35] Stallings, W. *Wireless Communications and Networking*, 1st Edition ed.; Prentice Hall: Upper Saddle River, NJ, 2002;.

- [36] Pesce, V.; Hermosin, P.; Rivolta, A.; Bhaskaran, S.; Silvestrini, S.; Colagrossi, A. Chapter Nine - Navigation. In *Modern Spacecraft Guidance, Navigation, and Control*; Pesce, V.; Colagrossi, A.; Silvestrini, S., Eds.; Elsevier: 2023; pp. 441-542.
- [37] Meng, Y.S.; Lee, Y.H. Study of shadowing effect by aircraft maneuvering for air-to-ground communication. *Aeu-international Journal of Electronics and Communications - AEU-INT J ELECTRON COMMUN* **2012**, 66, DOI 10.1016/j.aeue.2011.04.006.
- [38] Stellin, M.; Sabino, S.; Grilo, A. LoRaWAN Networking in Mobile Scenarios Using a WiFi Mesh of UAV Gateways. *Electronics* **2020**, 9, DOI 10.3390/electronics9040630.
- [39] Davoli, L.; Pagliari, E.; Ferrari, G. Hybrid LoRa-IEEE 802.11s Opportunistic Mesh Networking for Flexible UAV Swarming. *Drones* **2021**, 5, DOI 10.3390/drones5020026.
- [40] Suryadevara, N.K.; Dutta, A. Meshtastic Infrastructure-less Networks for Reliable Data Transmission to Augment Internet of Things Applications, Wireless and Satellite, , Systems; Guo, Q., Meng, W., Jia, M. and Wang, X., Eds.; Springer International Publishing: Cham, 2022; , pp. 622-640.
- [41] Lalle, Y.; Fourati, M.; Fourati, L.C.; Barraca, J.P. Routing Strategies for LoRaWAN Multi-Hop Networks: A Survey and an SDN-Based Solution for Smart Water Grid. *IEEE Access* **2021**, 9, 168624-168647, DOI 10.1109/ACCESS.2021.3135080. Available online: <https://ieeexplore.ieee.org/document/9648161>.
- [42] Diaz Zayas, A.; Merino, P. The 3GPP NB-IoT system architecture for the Internet of Things, - 2017 IEEE International Conference on Communications Workshops (ICC Workshops), 2017; , pp. 277-282.
- [43] 3GPP Releases. Available online: <https://www.3gpp.org/specifications/releases> (Accessed on 2 September 2022).
- [44] LoRaWAN L2 1.0.4 Specification. Available online: <https://resources.lora-alliance.org/technical-specifications/ts001-1-0-4-lorawan-l2-1-0-4-specification> (Accessed on 26 August 2022).
- [45] Lopez, V.H., Performance Evaluation of Long Range (LoRa) Wireless RF Technology for the Internet of Things (IoT) Using Dragino LoRa at 915 MHz. Master's Thesis, University of North Florida, Jacksonville, FL, USA, 2020.
- [46] AN1200.22 LoRa Modulation Basics. Available online: <https://www.semtech.com/products/wireless-rf/lora-connect/sx1276#documentation> (Accessed on 2 September 2022).
- [47] LoRa Connect. Available online: <https://www.semtech.com/products/wireless-rf/lora-connect#resources> (Accessed on 2 September 2022).
- [48] SX1276/77/78/79 Datasheet. Available online: <https://www.semtech.com/products/wireless-rf/lora-connect/sx1276#documentation> (Accessed on 2 September 2022).

- [49] SX1272/73 Datasheet. Available online: <https://www.semtech.com/products/wireless-rf/lora-connect/sx1272#documentation> (Accessed on 2 September 2022).
- [50] SX1268 Long Range, Low Power, sub-GHz RF Transceiver Datasheet. Available online: <https://www.semtech.com/products/wireless-rf/lora-connect/sx1268#documentation> (Accessed on 2 September 2022).
- [51] SX1261/2 Long Range, Low Power, sub-GHz RF Transceiver Datasheet. Available online: <https://www.semtech.com/products/wireless-rf/lora-connect/sx1261#documentation> (Accessed on 2 September 2022).
- [52] SX1280/SX1281 Datasheet. Available online: <https://www.semtech.com/products/wireless-rf/lora-connect/sx1280#documentation> (Accessed on 16 September 2022).
- [53] IEEE Standard for Information Technology--Telecommunications and Information Exchange between Systems - Local and Metropolitan Area Networks--Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. Available online: <https://ieeexplore.ieee.org/document/9363693> (Accessed on Mar 21, 2023).
- [54] IEEE Standard for Information technology-- Local and metropolitan area networks-- Specific requirements-- Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments. Available online: <https://ieeexplore.ieee.org/document/5514475> (Accessed on Mar 21, 2023).
- [55] IEEE 802.11s: The WLAN Mesh Standard. Available online: <https://ieeexplore.ieee.org/document/5416357> (Accessed on Mar 21, 2023).
- [56] IEEE Standard for Information Technology--Telecommunications and Information Exchange between Systems Local and Metropolitan Area Networks--Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 1: Enhancements for High-Efficiency WLAN. Available online: <https://ieeexplore.ieee.org/document/9442429> (Accessed on Mar 21, 2023).
- [57] Bouachir, O.; Abrassart, A.; Garcia, F.; Larrieu, N. A Mobility Model for UAV Ad Hoc Network, 2014 international conference on unmanned aircraft systems (ICUAS), IEEE: 2014; , pp. 383-388.
- [58] Nocedal, J.; Wright, S.J. *Numerical Optimization*, 2nd Edition ed.; Springer: New York, USA, 2006; pp. 2-9.
- [59] Hutchison, D.; Weikum, G.; Vardi, M.Y.; Tygar, D.; Terzopoulos, D.; Sudan, M.; Steffen, B.; Słowiński, R.; Pandu Rangan, C.; Nierstrasz, O.; Naor, M.; Mitchell, J.C.; Miettinen, K.; Mattern, F.; Kleinberg, J.M.; Kittler, J.; Kanade, T.; Deb, K.; Branke, J., Eds.; In *Multiobjective Optimization : Interactive and Evolutionary Approaches*; Lecture Notes in Computer Science; 2008; Volume 5252.
- [60] Kennedy, J.; Eberhart, R. Particle Swarm Optimization, Proceedings of ICNN'95 - International Conference on Neural Networks, 1995; , pp. 1942-1948 vol.4.

- [61] Engelbrecht, A.P. *Computational Intelligence: An Introduction*, 2nd Edition ed.; John Wiley & Sons: West Sussex, England, 2007; pp. 289-358.
- [62] Gallego-Madrid, J.; Molina-Zarca, A.; Sanchez-Iborra, R.; Bernal-Bernabe, J.; Santa, J.; Ruiz, P.M.; Skarmeta-Gómez, A.F. Enhancing Extensive and Remote LoRa Deployments through MEC-Powered Drone Gateways. *Sensors* **2020**, *20*, DOI 10.3390/s20154109.
- [63] Bravo-Arrabal, J.; Toscano-Moreno, M.; Fernandez-Lozano, J.; Mandow, A.; Gomez-Ruiz, J.; García-Cerezo, A. The Internet of Cooperative Agents Architecture (X-IoCA) for Robots, Hybrid Sensor Networks, and MEC Centers in Complex Environments: A Search and Rescue Case Study. *Sensors* **2021**, *21*, DOI 10.3390/s21237843.
- [64] Lakshmi, P.; Rejith, G.; Toby, T.; Sai Shibu, N.B.; Rao, S.N. A Resilient IoT System Architecture for Disaster Management in Collapsed Buildings, 2022; , pp. 282-287.
- [65] Marchese, M.; Moheddine, A.; Patrone, F. UAV and Satellite Employment for the Internet of Things Use Case, 2020; , pp. 1-8.
- [66] Sharma, R.; Arya, R. UAV based long range environment monitoring system with Industry 5.0 perspectives for smart city infrastructure. *Comput Ind Eng* **2022**, *168*, 108066, DOI <https://doi.org/10.1016/j.cie.2022.108066>. Available online: <https://www.sciencedirect.com/science/article/pii/S036083522200136X>.
- [67] Mujumdar, O.; Celebi, H.; Guvenc, I.; Sichitiu, M.; Hwang, S.; Kang, K. Use of LoRa for UAV Remote ID with Multi-User Interference and Different Spreading Factors, 2021; , pp. 1-7.
- [68] Delafontaine, V.; Schiano, F.; Cocco, G.; Rusu, A.; Floreano, D. Drone-aided Localization in LoRa IoT Networks, 2020; , pp. 286-292.
- [69] Rahman, G.M.E.; Wahid, K.A. LDAP: Lightweight Dynamic Auto-Reconfigurable Protocol in an IoT-Enabled WSN for Wide-Area Remote Monitoring. *Remote Sensing* **2020**, *12*, DOI 10.3390/rs12193131.
- [70] Bautista, O.; Akkaya, K.; Uluagac, A.S. Customized novel routing metrics for wireless mesh-based swarm-of-drones applications. *Internet of Things* **2020**, *11*, 100265, DOI <https://doi.org/10.1016/j.iot.2020.100265>. Available online: <https://www.sciencedirect.com/science/article/pii/S2542660520300998>.
- [71] Esrafilian, O.; Gangula, R.; Gesbert, D. Autonomous UAV-aided Mesh Wireless Networks, IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2020; , pp. 634-640.
- [72] Stellin, M., LoRa Networking in Mobile Scenarios using UAV Gateways. Master's Thesis, Instituto Superior Técnico, Lisboa, Portugal, 2018.



- [73] Xianfeng Li; Tao Zhang; Jianfeng Li A Particle Swarm Mobility Model for Flying Ad Hoc Networks, GLOBECOM 2017 - 2017 IEEE Global Communications Conference, , 2017IEEE: Singapore, 2017; , pp. 1-6.
- [74] Na, H.J.; Yoo, S. PSO-based dynamic UAV positioning algorithm for sensing information acquisition in wireless sensor networks. *Access* **2019**, *7*, 77499-77513, DOI 10.1109/ACCESS.2019.2922203. Available online: <https://ieeexplore.ieee.org/document/8735725>.
- [75] Pan, M.; Chen, C.; Yin, X.; Huang, Z. UAV-Aided Emergency Environmental Monitoring in Infrastructure-Less Areas: LoRa Mesh Networking Approach. *IEEE Internet of Things Journal* **2022**, *9*, 2918-2932, DOI 10.1109/JIOT.2021.3095494. Available online: <https://ieeexplore.ieee.org/document/9477431>.
- [76] Zirak, Q.; Shashev, D.; Shidlovskiy, S. Swarm of Drones Using LoRa Flying Ad-Hoc Network, 2021 International Conference on Information Technology (ICIT), 2021; , pp. 400-405.
- [77] Tanenbaum, A.S. *Computer Networks*, 5th Edition ed.; Prentice-Hall: Boston, Massachusetts, 2011; pp. 362-363.
- [78] GitHub - nohd0/Dijkstras: MATLAB implementation of Dijkstra's Algorithm. Available online: <https://github.com/nohd0/Dijkstras> (Accessed on 20 March 2023).
- [79] Wolfram Alpha. Available online: <https://www.wolframalpha.com/examples/mathematics/calculus-and-analysis/integrals> (Accessed on Apr 4, 2023).
- [80] Particle Swarm Optimization Algorithm. Available online: <https://www.mathworks.com/help/gads/particle-swarm-optimization-algorithm.html> (Accessed on Sep 2, 2022).
- [81] International Telecommunications Union, R.S. Report ITU-R M.2135-1: Guidelines for evaluation of radio interface technologies for IMT-Advanced. *International Telecommunications Union, Radiocommunications Sector* **2009**.
- [82] Small Unmanned Aircraft Systems (UAS) Regulations (Part 107). Available online: <https://www.faa.gov/newsroom/small-unmanned-aircraft-systems-uas-regulations-part-107> (Accessed on Sep 2, 2022).
- [83] van den Bergh, F.; Engelbrecht, A.P. A Study of Particle Swarm Optimization Particle Trajectories. *Inf Sci* **2006**, *176*, 937-971, DOI 10.1016/j.ins.2005.02.003. Available online: <https://www.sciencedirect.com/science/article/pii/S0020025505000630>.

## APPENDIX A: Main MATLAB Script

```
%% UAV_SWARM main script
%% Receive input parameters and initialize common variables (UAV, WLAN)
simTime = 40; % in seconds
updateRate = 2; % in Hz for UAV scenario and lidar sensors
time_step = 1; % in s for comms and mobility
FixedTRJHeight = 60; % in m
MaxHeight = 121; % in m
VertThresh = -82; % in dBm
LoRaThresh = -105; % in dBm
UAVinitspeed = 30; % in m/s
UAVmaxspeed = 44.5; % in m/s
UAVmaxaccel = 5; % in m/s2
gridsize = input("Grid size [m]:");
N = input("Number of drones:");
n = input("Number of ground nodes:");
PSO_type = input("PSO type: [1 for Fixed, 2 for Hybrid, 3 for PSO-only]:");
runflightsim = input("Run flight simulation? [1 for true, 0 for false]:");
runnetsim = input("Run network simulation? [1 for true, 0 for false]:");
switch runnetsim
    case 1
        % Show live state transition plot for all nodes
        showLiveStateTransitionPlot = input("Show Live State Transition Plot? [1 for true, 0 for false]:");
    otherwise
        showLiveStateTransitionPlot = false;
end
propaga = input("Propagation model [1 for Free Space, 2 for Ray Tracing, 3 for Log-normal 4 for TGax Residential]:");
backhaul = input("Backhaul: [1 for Wi-Fi, 2 for LoRa]:");
switch backhaul
    case 2
        freqLoRa = input("LoRa frequency: [1 for 900 MHz, 2 for 2.4 GHz, 3 for 400 MHz]:");
        BckhaulThresh = LoRaThresh; % in dBm
    otherwise
        freqLoRa = 0;
        BckhaulThresh = VertThresh; % in dBm
end
pfun_Eval = input("Apply DistanceNodesGain to: [1 path_exists, 2 More nodes + path_exists, 3 No path_exists, 4 Extra for other's paths, 5 Original, 6 More nodes]:");
pen_UAV1_CtrlSta = input("Reward/Penalize UAV1-to-CtrlSta link: [1 for true, 0 for false]:");
pen_exclo_a2g = input("Penalize if too close to the ground: [1 for true, 0 for false]:");
stopCriteria = input("Stopping Criteria: [1 for Mean Max, 2 for Mean Sum, 3 for Mean Pos]:");
switch freqLoRa
    case 0
        freqLoRa = 0;
    case 1
        freqLoRa = 900e6; % in Hz
    case 2
```



```

        freqLoRa = 2400e6; % in Hz
    case 3
        freqLoRa = 400e6; % in Hz
end

%% Create Scenario with Polygon Building Meshes

% Create the UAV scenario.
scene =
uavScenario("UpdateRate",updateRate,"StopTime",simTime,"Referencelocation",[30.27 -
81.505 0],"MaxNumFrames",200);

% Add a ground plane based on gridsize. East is X and North is Y.
color.Gray = 0.651*ones(1,3);
color.Green = [0.3922 0.8314 0.0745];
color.Red = [1 0 0];
color.Sky = [135/255 206/255 235/255];
addMesh(scene,"polygon",{[-gridsize/2 -gridsize/2; gridsize/2 -gridsize/2; ...
    gridsize/2 gridsize/2; -gridsize/2 gridsize/2], [-10 0]}, color.Gray)
addMesh(scene,"surface",{[gridsize/2 gridsize/2; gridsize/2 gridsize/2], ...
    [-gridsize/2 gridsize/2; -gridsize/2 gridsize/2], [0 0; 90 90]}, color.Sky)
addMesh(scene,"surface",{[gridsize/2 -gridsize/2; gridsize/2 -gridsize/2], ...
    [-gridsize/2 -gridsize/2; -gridsize/2 -gridsize/2], [0 0; 90 90]}, color.Sky)

% Show the scenario.
% Get screen resolution
resolution = get(0, 'screensize');
screenWidth = resolution(3);
screenHeight = resolution(4);
figureWidth = screenWidth*0.8;
figureHeight = screenHeight*0.8;
f1 = figure('Position', [screenWidth*0.025, screenHeight*0.075, figureWidth,
figureHeight]);
t = tiledlayout(f1,1,2);
ax = nexttile(t);
show3D(scene,'Parent',ax);
xlim([-250 250])
ylim([-250 250])
zlim([0 120])
view([-110 30])
axis equal
hold on

%% Set starting and ending positions, create UAV Fixed trajectories, create UAV
platforms, and mount sensors (UAV TOOLBOX)
nlength = floor(sqrt(N));
elength = ceil(N/nlength);

% initial_pose = [gridsize/2 -gridsize/2 0 1 0 0 0];
startPos = repmat([-gridsize/2 gridsize/2 0],1,1,nlength*elength);

% Create destination positions based on gridsize, the number of drones N
% and a 120 m height. Height is POSITIVE here because of the ENU
% reference frame for UAVs and sensors.
cellcount = 1;

```

```

destPos = zeros(1,3,nlength*elength);
eucdistance_vec = [];
for column = 1:nlength
    for row = 1:elength
        destPos(1,:,cellcount) = [-(gridsize/2)-
(gridsize/elength/2)+row*(gridsize/elength),...
        (gridsize/2)+(gridsize/nlength/2)-column*(gridsize/nlength),
FixedTRJHeight];
        eucdistance_vec = cat(3,eucdistance_vec,sqrt((destPos(1,1,cellcount)-
startPos(1,1,cellcount))^2+ ...
        (destPos(1,2,cellcount)-
startPos(1,2,cellcount))^2+(destPos(1,3,cellcount)-startPos(1,3,cellcount))^2));
        cellcount = cellcount + 1;
    end
end

% Create Waypoints vectors, orientations and ToAs for each UAV: 2 wp, 2 orient and 1
% ToA for each UAV. ToA is based on s-d distances and constant speed
% (calculated from the distance to the farthest destination and simTime

waypoints = [startPos;destPos];
orientation_eul = [0 0 0];
orientation_quat = quaternion(eul2quat(orientation_eul));
orientation_vec = repmat(orientation_quat,size(waypoints,1),1,nlength*elength);
speed = max(eucdistance_vec)/simTime;
toa_vect = eucdistance_vec/UAVinitspeed;
FixedTRJTime = max(eucdistance_vec)/UAVinitspeed;
sampleTimes = 0:time_step:FixedTRJTime;

% Initialize platforms and sensors to improve runtime
plat = uavPlatform.empty;
lidar = uavSensor.empty;

% Initialize trajectory elements arrays
trajectory = cell(1,1,nlength*elength);
pos_array = zeros(500,3,N);
ori_array = zeros(500,1,N,'quaternion');
vel_array = zeros(500,3,N);
acc_array = zeros(500,3,N);
ang_array = zeros(500,3,N);

% Initialize lidar model
lidarmodel1 = uavLidarPointCloudGenerator("AzimuthResolution",0.3324099,...
    "ElevationLimits",[-90 20],"ElevationResolution",1.25,...
    "MaxRange",120,"UpdateRate",2,"HasOrganizedOutput",true);

% Iterate to create trajectories for each cell
for cellcount = 1:nlength*elength
    % Create Trajectory based on starting and final positions
    trajectory{1,1,cellcount} =
    waypointTrajectory("Waypoints",waypoints(:, :, cellcount),...
        "Orientation",orientation_vec(:, :, cellcount),"SampleRate",(1/time_step),...
        "ReferenceFrame","ENU","TimeOfArrival",[0 toa_vect(1,1,cellcount)]);
end

```

```

    % Look up the waypoints of the trajectory in the local navigation coordinate
    system in meters.
    [position,orientation,velocity,acceleration,angularVelocity] =
    lookupPose(trajectory{1,1,cellcount},sampleTimes);

    % Clean trajectories of NaN (replace NaN with the last position).
    traject_length = size(position,1);
    for traj_count = 1:traject_length
        if anynan(position(traj_count,:))
            position(traj_count,:) = position(traj_count-1,:);
            orientation(traj_count,:) = orientation(traj_count-1,:);
            velocity(traj_count,:) = velocity(traj_count-1,:);
            acceleration(traj_count,:) = acceleration(traj_count-1,:);
            angularVelocity(traj_count,:) = angularVelocity(traj_count-1,:);
        end
    end

    % Populate trajectory elements arrays
    pos_array(1:size(position,1),:,cellcount) = position;
    ori_array(1:size(position,1),:,cellcount) = orientation;
    vel_array(1:size(position,1),:,cellcount) = velocity;
    acc_array(1:size(position,1),:,cellcount) = acceleration;
    ang_array(1:size(position,1),:,cellcount) = angularVelocity;
end

% Set up platforms at their initial positions to be used together with
% the move method
for platcount = 1:N
    UAVstrname = strcat("UAV",string(platcount));
    plat(1,1,platcount) = uavPlatform(UAVstrname,scene,"ReferenceFrame","ENU",...
        "InitialPosition",pos_array(1,:,platcount),...
        "InitialVelocity",vel_array(1,:,platcount),...
        "InitialAcceleration",acc_array(1,:,platcount),...
        "InitialOrientation",eul2quat(quat2eul(ori_array(1,:,platcount))),...
        "InitialAngularVelocity",ang_array(1,:,platcount));

    % Set up platform mesh. Add a rotation to orient the mesh to the UAV body frame.
    updateMesh(plat(1,1,platcount),"quadrotor",{10},color.Red,[0 0 0],eul2quat([0 0
    pi]));

    % Mount sensors. You can choose to mount different sensors to your UAV
    %SENSORstrname = strcat("Lidar",string(cellcount));
    lidar(1,1,platcount) =
    uavSensor("Lidar",plat(1,1,platcount),lidarmodel1,"MountingLocation",[0 0 -
    1],"MountingAngles",[0 0 0]);
end

%% Create WiFi Network
% Create APs and Ground Nodes (WLAN TOOLBOX modified)

% Configuration Parameters
rng(1, 'simdTwister'); % Seed for random number generator
displayStatistics = false; % Display statistics at the end of the
simulation

```

```

% Add the folder to the path for access to all helper files
addpath(genpath(fullfile(pwd, 'm1WLANSystemSimulation')));

ScenarioParameters = struct;
% Number of rooms in [x,y,z] directions
ScenarioParameters.BuildingLayout = [1 1 1];
% Size of each room in meters [x,y,z]
ScenarioParameters.RoomSize = [gridsize gridsize MaxHeight+100];
% Number of STAs per room
ScenarioParameters.NumRxPerRoom = n;

% Drop nodes
staPositions = hDropNodes(n, gridsize);
interm = permute(pos_array(1,1:3,:),[3 2 1]);
apPositions = interm(:, :, 1);
staPositions = cat(1, staPositions, [(-gridsize/2)*1 (gridsize/2)*1 1.5]);

% Create triangulation object for TGax propagation model
tri = hTGaxResidentialTriangulation(ScenarioParameters);

% Set propagation model
switch propaga
case 1
    propModel =
hFreeSpacePathLoss('Triangulation',tri,'ShadowSigma',0,'FacesPerWall',1);
    if backhaul == 2
        propModelHori = propagationModel('freespace');
    end
case 2
    propModel =
propagationModel("raytracing","Method","image","MaxNumReflections",1,"CoordinateSystem","cartesian");
    if backhaul == 2
        propModelHori =
propagationModel("raytracing","Method","image","MaxNumReflections",1,"CoordinateSystem","cartesian");
    end
case 3
    propModel = propagationModel('close-in');
    propModel.PathLossExponent = 2.2;
    propModel.Sigma = 0.1;
    if backhaul == 2
        propModelHori = propagationModel('close-in');
        propModelHori.PathLossExponent = 2.2;
        propModelHori.Sigma = 0.1;
    end
case 4
    propModel =
hTGaxResidentialPathLoss('Triangulation',tri,'ShadowSigma',0.1,'FacesPerWall',1);
    if backhaul == 2
        propModelHori = propagationModel('close-in');
        propModelHori.ReferenceDistance = 5;
        propModelHori.PathLossExponent = 3.5;
        propModelHori.Sigma = 0.1;
    end
end

```

```

end

% Node parameters
% Get the IDs, positions, and traffic configurations of each node
switch backhaul
    case 2
        [nodeConfigs, trafficConfigs] = hLoadConfigurationFull_Int_Traff_7(N, n,
apPositions, staPositions, VertThresh);
    otherwise
        [nodeConfigs, trafficConfigs] = hLoadConfigurationFull_Int_Traff_6(N, n,
apPositions, staPositions, VertThresh);
end

% Set abstraction level
MACFrameAbstraction = true;
PHYAbstractionType = "TGax Evaluation Methodology Appendix 1";

disablenames = false;

%pl = zeros(N+n,N+n,size(pos_array, 1));
TxPowerMat = zeros(1,N+n+1);
TxAntGMat = zeros(1,N+n+1);
RxAntGMat = zeros(1,N+n+1);
for wcount = 1:(N+n+1)
    TxAntGMat(1,wcount) = nodeConfigs(wcount).TxGain;
    RxAntGMat(1,wcount) = nodeConfigs(wcount).RxGain;
end
TxAntGMat = repmat(TxAntGMat,N+n+1,1);
RxAntGMat = repmat(RxAntGMat,N+n+1,1);
RxPowerMat = zeros(N+n+1,N+n+1,size(pos_array, 1));
RxPowerMatW = zeros(N+n+1,N+n+1,size(pos_array, 1));
RxPowerMatDij = zeros(N+n+1,N+n+1);

%% Fly the UAV Platforms Along the Fixed Trajectories and Collect Point Cloud Sensor
Readings

% Visualize the scene
figure(f1)
[ax,plotFrames] = show3D(scene);

% Update plot view for better visibility
xlim([-250 200])
ylim([-150 180])
zlim([0 50])
view([-110 20])
axis equal
hold on

% Create a scatter plot for the point clouds. Update the data source properties
again.
colormap("jet")
pt = pointCloud(nan(N,1,3));
% Uncomment and repeat the following code as many times as there are UAVs
% scatterplot1 = scatter3(ax,nan,nan,nan,1,[0.3020 0.7451 0.9333],...
%     "Parent",plotFrames.UAV1.Lidar);

```

```

% scatterplot1.XDataSource = "reshape(pt(1,1).Location(:,:,1),[],1)";
% scatterplot1.YDataSource = "reshape(pt(1,1).Location(:,:,2),[],1)";
% scatterplot1.ZDataSource = "reshape(pt(1,1).Location(:,:,3),[],1)";
% scatterplot1.CDataSource = "reshape(pt(1,1).Location(:,:,3),[],1) -
min(reshape(pt(1,1).Location(:,:,3),[],1))";
% scatterplot2 = scatter3(nan,nan,nan,1,[0.3020 0.7451 0.9333],...
%     "Parent",plotFrames.UAV2.Lidar);

%Create an occupancy map for a more efficient way to store the point cloud
%data. Use a minimum resolution of 1 cell per meter.
map3D = occupancyMap3D(1);

% Set up the simulation
setup(scene)

% Initialize sensor variables, lastest-position vectors, and PSO variables
ptIdx = 1;
isupdated = zeros(1,1,N);
lidarSampleTime = zeros(1,1,N);
motion = zeros(size(pos_array, 1),16,N);    % read(plat) returns a 16-element motion
vector
LLA = zeros(size(pos_array, 1),3,N);        % read(plat) returns a 3-element LLA
vector
cont = 0;
pfun = 0;
fBest = zeros(size(pos_array, 1),N);
pfBest = zeros(size(pos_array, 1),N);
clusterfBest = zeros(size(pos_array, 1),N,N);
pBestInst = zeros(1,N);
pBestPos = pos_array(1,:,:);
gBestPos = zeros(N,3);
if PSO_type == 3
    fixedTraj = 1;
else
    fixedTraj = size(position, 1);
end
itera = true;
w = 0.95*ones(1,N);
y1 = 1.35;
y2 = 0.01;
stallcount = zeros(1,N);
flag = false;
yy = zeros(N,ptIdx);
labels = [];
g1 = gobjects(N,N);
g2 = gobjects(N,n);
g3 = gobjects(1,1);
plr = zeros(N+n+1,N+n+1);

for platcount = 1:N
    labels = cat(2,labels,strcmp('UAV',string(platcount)));
end

% Iterate through the positions and show the scene each time any of the lidar sensors
updates

```

```

while itera

    delete(g1(:,,:));
    delete(g2(:,,:));
    delete(g3(:,,:));

    path_exists = zeros(1,N+n);

    % Read all platforms' latest position and
    % Rearrange pos_array to accommodate WLAN reqs.
    for platcount = 1:N
        [motion(ptIdx,:,platcount),LLA(ptIdx,:,platcount)] =
read(plat(:, :, platcount));
    end
    interm = permute(motion(ptIdx,1:3,:),[3 2 1]);
    apPositions = interm(:, :, 1);

    % Simulate wireless network
    % Get updated AP positions
    if ptIdx > 1
        for platcount = 1:N
            nodeConfigs(platcount).NodePosition = apPositions(platcount,:);
        end
    end

    % Create transmitter and receiver sites
    [txs,rxs] = hCreateSitesFromNodesDP(nodeConfigs,N,n);
    switch freqLoRa
        case 0
        otherwise
            %txLoRa = txs;
            %rxLoRa = rxs;
            [txLoRa,rxLoRa] = hCreateSitesFromNodesDP(nodeConfigs,N,n);
            for i = 1:N
                txLoRa(i).TransmitterFrequency = freqLoRa;
            end
        end

    % Visualize the scenario
    [Txnameshandle,Rxnameshandle,UAVshandle] =
hVisualizeScenarioDP(tri,ax,txs,rxs,apPositions,"DisableNames",disablenames);

    % Obtain pathloss and RxPower between each pair of nodes
    for wcount = 1:(N+n+1)
        TxPowerMat(1,wcount) = nodeConfigs(wcount).TxPower;    %Allows dynamic power
    end
    TxPowerMat = repmat(TxPowerMat,N+n+1,1);
    switch propaga
        case 2
            [plWifi,pathlossFnHdl] = hCreatePathlossTableDP(txs,rxs,propModel,1);
            if backhaul == 2
                plLoRa = pathloss(propModelHori,rxLoRa,txLoRa);
                for row = 1:N
                    for column = 1:N

```

```

        if plLoRa{row,column} ~= -Inf
            plr(row,column) = [plLoRa{row,column}];
        else
            plr(row,column) = 0;
        end
    end
end
end
otherwise
    [plWiFi,pathlossFnHdl] = hCreatePathlossTableDP(txs,rxs,propModel);
    if backhaul == 2
        plLoRa = pathloss(propModelHori,rxLoRa,txLoRa);
        for row = 1:N
            for column = 1:N
                if plLoRa(row,column) ~= -Inf
                    plr(row,column) = plLoRa(row,column);
                else
                    plr(row,column) = 0;
                end
            end
        end
    end
end
switch backhaul
    case 2
        RxPowerMat(1:N,1:N,ptIdx) = TxPowerMat(1:N,1:N,1) + TxAntGMat(1:N,1:N,1)
- plr(1:N,1:N,1);
        RxPowerMat(N+1:N+n+1,1:N+n+1,ptIdx) = TxPowerMat(N+1:N+n+1,1:N+n+1,1) +
TxAntGMat(N+1:N+n+1,1:N+n+1,1) - plWiFi(N+1:N+n+1,1:N+n+1,1);
        RxPowerMat(1:N,N+1:N+n+1,ptIdx) = TxPowerMat(1:N,N+1:N+n+1,1) +
TxAntGMat(1:N,N+1:N+n+1,1) - plWiFi(1:N,N+1:N+n+1,1);
    otherwise
        RxPowerMat(:, :, ptIdx) = TxPowerMat + TxAntGMat - plWiFi(:, :, 1);
    end
    RxPowerMatW(:, :, ptIdx) = 10.^(RxPowerMat(:, :, ptIdx)/10);
    RxPowerMatDij = RxPowerMat(:, :, ptIdx);
    RxPowerMatDij = RxPowerMatDij - diag(diag(RxPowerMatDij));
    for row = 1:N
        for column = 1:N
            if RxPowerMatDij(row,column) >= BckhaulThresh
                RxPowerMatDij(row,column) = 1;
            else
                RxPowerMatDij(row,column) = Inf;
            end
        end
    end
    for row = N+1:N+n+1
        for column = 1:N+n+1
            if RxPowerMatDij(row,column) >= VertThresh
                RxPowerMatDij(row,column) = 1;
            else
                RxPowerMatDij(row,column) = Inf;
            end
        end
    end
end
end

```



```

for row = 1:N
    for column = N+1:N+n+1
        if RxPowerMatDij(row,column) >= VertThresh
            RxPowerMatDij(row,column) = 1;
        else
            RxPowerMatDij(row,column) = Inf;
        end
    end
end
RxPowerMatDijUAV = RxPowerMatDij;
RxPowerMatDijUAV(N+1:N+n,:) = Inf;
RxPowerMatDijUAV(:,N+1:N+n) = Inf;
TxPowerMat = zeros(1,N+n+1);

%% Determine if a path to ControlSTA exists

% Interface index on which packet has to be forwarded to next node.
destID = N+n+1;      % Destination node ID (ControlSTA)

% Configure routing table at MeshN to reach ControlSTA
% MeshN-1 is the next hop node from MeshN
for platcount = 1:N
    [cost, Dij_path] = Dijkstras(RxPowerMatDijUAV,platcount,destID);
    if cost ~= Inf
        path_exists(1,platcount) = 1;
    end
end

%% Particle Swarm Optimization

% Evaluate objective function, get best p and g values, and get best group
position
% Move through UAVs
for platcount = 1:N
    [pfun,g1,g2,g3] =
ObjFunEval(RxPowerMatW,VertThresh,BckhaulThresh,path_exists,...
N,n,ptIdx,platcount,ax,g1,g2,g3,pos_array,staPositions,gridsize,pfun_Eval,...
    pen_exclo_a2g,pen_UAV1_CtrlSta);
    %fprintf('2. %d %d %.2d %d %d\n', ptIdx, platcount, pfun, 10*log10(pfun),
minTemp)
    if ptIdx > 1
        if pfun > pfBest(ptIdx-1,platcount)
            pfBest(ptIdx,platcount) = pfun;
            pBestInst(1,platcount) = ptIdx;
            pBestPos(1,:,platcount) = motion(ptIdx,1:3,platcount);    %Get best
individual positions
        else
            pfBest(ptIdx,platcount) = pfBest(ptIdx-1,platcount);
        end
    else
        if pfun > pfBest(ptIdx,platcount)
            pfBest(ptIdx,platcount) = pfun;
            pBestInst(1,platcount) = ptIdx;

```

```

        pBestPos(1,:,platcount) = motion(ptIdx,1:3,platcount);    %Get best
individual positions
    end
end

% Draw lidar plots to simulate coverage
% This part must be located after the part that plots the link
% lines (ObjFunEval) so they can be seen before being deleted in the
% next iteration, but before the loop that moves the platforms. That is
% why this code is in the middle of the PSO section.
% Read all sensors' data from the scenario
for lidarcount = 1:N
    [isupdated(1,1,lidarcount),lidarSampleTime(1,1,lidarcount), pt(lidarcount,:)]
= read(lidar(1,1,lidarcount));
end

if runflightsim
    % If any sensor is updated then show the scene
    if any(isupdated)
        % Use fast update to move platform visualization frames.

show3D(scene,"Time",lidarSampleTime(1,1,N),"FastUpdate",true,"Parent",ax);
        % Refresh all plot data and visualize.
        refreshdata(f1,'caller')
        drawnow
    end
end

% Determine GBest locally at each UAV among those UAVs within its range
% and move platforms.
for platcount = 1:N
    % Determine GBest locally at each UAV among those UAVs within its
    % range (cluster/neighborhood).
    for pfBestcount =1:N
        if pfBestcount ~= platcount %Not considering itself for cluster best
            if RxPowerMat(pfBestcount,platcount,ptIdx) >= BckhaulThresh
                clusterfBest(ptIdx,pfBestcount,platcount) =
pfBest(ptIdx,pfBestcount);
            else
                clusterfBest(ptIdx,pfBestcount,platcount) = 0;
            end
        else
            clusterfBest(ptIdx,pfBestcount,platcount) = 0;
        end
    end
    [fBestTemp, gBestNodeTemp] = max(clusterfBest(ptIdx,:,platcount));
    if ptIdx > 1
        if ptIdx >= fixedTraj
            if fBestTemp > fBest(ptIdx-1,platcount)
                fBest(ptIdx,platcount) = fBestTemp;
                gBestInst = ptIdx;
                gBestNode = gBestNodeTemp;
                gBestPos(platcount,:) = pBestPos(1,:,gBestNode);
                stallcount(1,platcount) = max(0, stallcount(1,platcount)-1);
            end
        end
    end
end

```

```

        if stallcount(1,platcount) < 2
            w(1,platcount) = min(2*w(1,platcount),UAVmaxspeed);
        end
        if stallcount(1,platcount) > 5
            w(1,platcount) = w(1,platcount)/2;
        end
    else
        stallcount(1,platcount) = stallcount(1,platcount)+1;
        fBest(ptIdx,platcount) = fBest(ptIdx-1,platcount);
    end
else
    if fBestTemp > fBest(ptIdx-1,platcount)
        fBest(ptIdx,platcount) = fBestTemp;
        gBestInst = ptIdx;
        gBestNode = gBestNodeTemp;
        gBestPos(platcount,:) = pBestPos(1,:,gBestNode);
    else
        fBest(ptIdx,platcount) = fBest(ptIdx-1,platcount);
    end
end
else
    if fBestTemp > fBest(ptIdx,platcount)
        fBest(ptIdx,platcount) = fBestTemp;
        gBestInst = ptIdx;
        gBestNode = gBestNodeTemp;
        gBestPos(platcount,:) = pBestPos(1,:,gBestNode);
    end
end

% Move platforms.
if ptIdx < fixedTraj
    move(plat(1,1,platcount),[pos_array(ptIdx+1,:,platcount),...
        vel_array(ptIdx+1,:,platcount),...
        acc_array(ptIdx+1,:,platcount),...
        eul2quat(quat2eul(ori_array(ptIdx+1,:,platcount))),...
        ang_array(ptIdx+1,:,platcount)])
else
    if PSO_type ~= 1
        vel_array(ptIdx+1,:,platcount) =
w(1,platcount)*vel_array(ptIdx,:,platcount) + ...
        y1*rand()*(pBestPos(1,:,platcount)-pos_array(ptIdx,:,platcount))
+ ...
        y2*rand()*(gBestPos(platcount,:)-pos_array(ptIdx,:,platcount));
        speed2 = sqrt(vel_array(ptIdx+1,1,platcount)^2 +
vel_array(ptIdx+1,2,platcount)^2 + ...
        vel_array(ptIdx+1,3,platcount)^2);
        if speed2 > UAVmaxspeed
            vel_array(ptIdx+1,:,platcount) = vel_array(ptIdx+1,:,platcount) *
...
            UAVmaxspeed / speed2;
        end
        diffpos = time_step*vel_array(ptIdx+1,:,platcount);
        nextpos = pos_array(ptIdx,:,platcount) + diffpos;
        if nextpos(1,1,1) >= -gridsize/2 && nextpos(1,1,1) <= gridsize/2 &&
...

```

```

        nextpos(1,2,1) >= -gridsize/2 && nextpos(1,2,1) <= gridsize/2
&& ...
        nextpos(1,3,1) >= 0 && nextpos(1,3,1) <= MaxHeight
pos_array(ptIdx+1,:,platcount) = pos_array(ptIdx,:,platcount) +
diffpos;

        acc_array(ptIdx+1,:,platcount) = acc_array(ptIdx,:,platcount);
        ori_array(ptIdx+1,:,platcount) = ori_array(ptIdx,:,platcount);
        ang_array(ptIdx+1,:,platcount) = ang_array(ptIdx,:,platcount);
    else
        if nextpos(1,1,1) < -gridsize/2 || nextpos(1,1,1) > gridsize/2
            vel_array(ptIdx+1,1,platcount) = 0;
        end
        if nextpos(1,2,1) < -gridsize/2 || nextpos(1,2,1) > gridsize/2
            vel_array(ptIdx+1,2,platcount) = 0;
        end
        if nextpos(1,3,1) < 0 || nextpos(1,3,1) > MaxHeight
            vel_array(ptIdx+1,3,platcount) = 0;
        end
        diffpos = time_step*vel_array(ptIdx+1,:,platcount);
        pos_array(ptIdx+1,:,platcount) = pos_array(ptIdx,:,platcount) +
diffpos;

        acc_array(ptIdx+1,:,platcount) = acc_array(ptIdx,:,platcount);
        ori_array(ptIdx+1,:,platcount) = ori_array(ptIdx,:,platcount);
        ang_array(ptIdx+1,:,platcount) = ang_array(ptIdx,:,platcount);
    end
    move(plat(1,1,platcount),[pos_array(ptIdx+1,:,platcount),...
        vel_array(ptIdx+1,:,platcount),...
        acc_array(ptIdx+1,:,platcount),...
        eul2quat(quat2eul(ori_array(ptIdx+1,:,platcount))),...
        ang_array(ptIdx+1,:,platcount)])
    else
        itera = false;
    end
end
% Create a line plot for the trajectories
if (PSO_type == 1 || PSO_type == 2) && itera == true
    plot3(ax,pos_array(ptIdx:ptIdx+1,1,platcount),...
        pos_array(ptIdx:ptIdx+1,2,platcount),...
        pos_array(ptIdx:ptIdx+1,3,platcount),...
        "Color",[1 1 1],"LineWidth",2);
end

% Create a tile within the figure to plot the RxPowerLevels
% It only happens during the first iteration
if and(ptIdx == 1,platcount ==1)
    ax2 = nexttile(t);
end
end

% Create a line plot for the RxPowerLevels (depends on the selected
% stopping criteria
switch stopCriteria
    case 1
        pivotmat2 = reshape(max(RxPowerMat(N+1:N+n,1:N,ptIdx),[],1),1,[]);
    otherwise

```

```

        pivotmat2 =
reshape(10.*log10(sum(RxPowerMatW(N+1:N+n,1:N,ptIdx),1)),1,[]);
    end
    yy(1:N,ptIdx) = pivotmat2;
    plot(ax2,1:ptIdx,yy(1:N,1:ptIdx),"LineWidth",1);
    title(ax2,'Received Power Levels at Each UAV')
    legend(ax2,labels)
    xlabel(ax2, 'Number of Iterations')
    ylabel(ax2, 'Received Power Levels [dBm]')
    set(ax2, 'YGrid', 'on', 'XGrid', 'off')

% Check if stopping criteria is met
switch stopCriteria
    case 1
        switch PSO_type
            case 3
                if ptIdx > 10 && ptIdx > fixedTraj
                    change = abs(max(RxPowerMat(N+1:N+n,1:N,ptIdx),[],1)-...
                        mean(max(RxPowerMat(N+1:N+n,1:N,ptIdx-10:ptIdx-
1),[],1),3)));
                    relatchange_mean =
change./abs(mean(max(RxPowerMat(N+1:N+n,1:N,ptIdx-10:ptIdx-1),[],1),3)));
                    if max(relatchange_mean) <= 0.02 || min(stallcount) >= 150 ||
ptIdx >= 180
                        itera = false;
                    end
                end
            otherwise
                if ptIdx > 10 && ptIdx > fixedTraj + 20
                    change = abs(max(RxPowerMat(N+1:N+n,1:N,ptIdx),[],1)-...
                        mean(max(RxPowerMat(N+1:N+n,1:N,ptIdx-10:ptIdx-
1),[],1),3)));
                    relatchange_mean =
change./abs(mean(max(RxPowerMat(N+1:N+n,1:N,ptIdx-10:ptIdx-1),[],1),3)));
                    if max(relatchange_mean) <= 0.02 || min(stallcount) >= 150 ||
ptIdx >= 180
                        itera = false;
                    end
                end
            end
        end
    case 2
        switch PSO_type
            case 3
                if ptIdx > 10 && ptIdx > fixedTraj
                    change = abs(sum(RxPowerMatW(N+1:N+n,1:N,ptIdx),1)-...
                        mean(sum(RxPowerMatW(N+1:N+n,1:N,ptIdx-10:ptIdx-
1),1),3)));
                    relatchange_mean =
change./abs(mean(sum(RxPowerMatW(N+1:N+n,1:N,ptIdx-10:ptIdx-1),1),3)));
                    if max(relatchange_mean) <= 0.02 || min(stallcount) >= 150 ||
ptIdx >= 180
                        itera = false;
                    end
                end
            otherwise

```

```

        if ptIdx > 10 && ptIdx > fixedTraj + 20
            change = abs(sum(RxPowerMatW(N+1:N+n,1:N,ptIdx),1)-...
                mean(sum(RxPowerMatW(N+1:N+n,1:N,ptIdx-10:ptIdx-
1),1),3));
            relatchange_mean =
change./abs(mean(sum(RxPowerMatW(N+1:N+n,1:N,ptIdx-10:ptIdx-1),1),3));
            if max(relatchange_mean) <= 0.02 || min(stallcount) >= 150 ||
ptIdx >= 180
                itera = false;
            end
        end
    end
    otherwise
        switch PSO_type
            case 3
                if ptIdx > 10 && ptIdx > fixedTraj
                    cent_grav = mean(pos_array(ptIdx-10:ptIdx-1,:,1:N),1); %
Center of gravity 10 previous pos
                    cent_grav = repmat(cent_grav,11,1);
                    pos_change = pos_array(ptIdx-10:ptIdx,:,1:N)-cent_grav; %
Distances from all 11 last points
                    pos_change = pos_change.^2;
                    pos_change = sum(pos_change,2).^0.5;
                    if max(mean(pos_change)) <= UAVmaxspeed*time_step ||
min(stallcount) >= 150 || ptIdx >= 180 %Mean inside sphere around CoG
                        itera = false;
                    end
                end
            end
        otherwise
            if ptIdx > 10 && ptIdx > fixedTraj + 20
                cent_grav = mean(pos_array(ptIdx-10:ptIdx-1,:,1:N),1); %
Center of gravity 10 previous pos
                cent_grav = repmat(cent_grav,11,1);
                pos_change = pos_array(ptIdx-10:ptIdx,:,1:N)-cent_grav; %
Distances from all 11 last points
                pos_change = pos_change.^2;
                pos_change = sum(pos_change,2).^0.5;
                if max(mean(pos_change)) <= UAVmaxspeed*time_step ||
min(stallcount) >= 150 || ptIdx >= 180 %Mean inside sphere around CoG
                    itera = false;
                end
            end
        end
    end
    ptIdx = ptIdx + 1;
    % Advance scene simulation time
    advance(scene);
    % Update all sensors in the scene.
    updateSensors(scene)
    % Delete UAVs names from plot
    if itera
        delete(Txnameshandle);
        delete(Rxnameshandle);
        delete(UAVshandle);
    end
end

```

```

end

if ~runflightsim
% If any sensor is updated then show the scene
    if any(isupdated)
        % Use fast update to move platform visualization frames.
        show3D(scene,"Time",lidarSampleTime(1,1,N),"FastUpdate",true,"Parent",ax);
        % Refresh all plot data and visualize.
        refreshdata(f1,'caller') % Esto hace que se actualicen las variables sin
        estar en el lazo
        drawnow
    end
end

%% Simulate communication
% Create nodes
wlanNodes = hCreateWLANNodes(nodeConfigs, trafficConfigs, ...
'CustomPathLoss', pathlossFnHdl, 'MACFrameAbstraction', MACFrameAbstraction,
'PHYAbstractionType', PHYAbstractionType);

% Configure mesh routing table
% Interface index on which packet has to be forwarded to next node.
forwardInterfaceID = 1;
destID = N+n+1; % Destination node ID (ControlSTA)
destAddress = wlanNodes{destID}.MAC.MACAddress; % Destination MAC address

for platcount = 1:N
    [cost, Dij_path] = Dijkstras(RxPowerMatDijUAV,platcount,destID);
    if cost ~= Inf
        nextHopAddress = wlanNodes{Dij_path(2)}.MAC.MACAddress; % Next hop MAC
        address
        addPath(wlanNodes{platcount}, destID, destAddress, nextHopAddress, ...
            forwardInterfaceID); %ForwardTable
        path_exists(1,platcount) = 1;
    end
end

for ground_node = N+1:N+n
    RxPowerMatDijGN = RxPowerMatDij;
    RxPowerMatDijGN(N+1:N+n,:) = Inf;
    RxPowerMatDijGN(:,N+1:N+n) = Inf;
    RxPowerMatDijGN(ground_node,1:N) = RxPowerMatDij(ground_node,1:N);
    RxPowerMatDijGN(1:N,ground_node) = RxPowerMatDij(1:N,ground_node);
    [cost, Dij_path] = Dijkstras(RxPowerMatDijGN,ground_node,destID);
    if cost ~= Inf
        nextHopAddress = wlanNodes{Dij_path(2)}.MAC.MACAddress; % Next hop MAC
        address
        addPath(wlanNodes{ground_node}, destID, destAddress, nextHopAddress,
            forwardInterfaceID); %ForwardTable
        path_exists(1,ground_node) = 1;
    end
end

% WiFi visualization parameters
% Initialize visualization parameters

```

```

visualizationInfo = struct;
visualizationInfo.Nodes = wlanNodes;
statsLogger = hWLANStatsLogger(visualizationInfo);
statistics = cell(0);

if runnetsim
    % Configure state transition visualization
    if showLiveStateTransitionPlot
        hPlotStateTransition(visualizationInfo);
    end

    % Initialize wireless network simulator
    networkSimulator = hWirelessNetworkSimulator(wlanNodes);

    % When you run the script from the MATLAB command prompt, pause the
    % execution to refresh visualization after every 5 milliseconds

    if showLiveStateTransitionPlot
        scheduleEvent(networkSimulator, @() pause(0.001), [], 0, 5);
    end
    run(networkSimulator, time_step*1000*1);

    % Retrieve the statistics
    statistics = getStatistics(statsLogger, displayStatistics);

    % Plot the throughput, packet loss ratio, and average packet latency at each node
    hPlotNetworkStats(statistics, wlanNodes);
end

pfBestdBm = 10.*log10(pfBest);

for nodecount = 1:N+n+1
    for instant = 1:ptIdx-1
        TableForExcel(instant,:,nodecount) = RxPowerMat(nodecount,:,instant);
    end
end

writematrix(TableForExcel,'RxPowerMat.xlsx')

% Cleanup the persistent variables used in functions
clear hPlotStateTransition;

% Save the statistics to a mat file
save('statistics.mat', 'statistics');

hold off

%% Objective Function Evaluation
function [pfun,g1,g2,g3] =
ObjFunEval(RxPowerMatW,VertThresh,BckhaulThresh,path_exists,...
N,n,ptIdx,platcount,ax,g1,g2,g3,pos_array,staPositions,gridsize,pfun_Eval,...
pen_exclo_a2g,pen_UAV1_CtrlSta)
pfun = 0;
cont = 0;

```



```

% Move through ground nodes searching for Rx powers greater than
% VertThresh dBm and add them up if more than one is found
for ground_node = N+1:N+n
    maxTemp = abs(RxPowerMatW(ground_node,platcount,ptIdx));
    if maxTemp >= 10^(VertThresh/10)
        cont = cont + 1;
        if cont > 1
            pfun = pfun + maxTemp;
        else
            pfun = maxTemp;
        end
        % Create a line plot for the connections between UAVs and
        % ground nodes
        g2(platcount, ground_node-N) = plot3(ax,[pos_array(ptIdx,1,platcount)
staPositions(ground_node-N,1)],...
        [pos_array(ptIdx,2,platcount) staPositions(ground_node-N,2)],...
        [pos_array(ptIdx,3,platcount) staPositions(ground_node-N,3)],...
        ':' ,"Color",[0 1 0] ,"LineWidth",0.75);
    else
        if maxTemp > pfun
            pfun = maxTemp;
        end
    end
end

switch pen_exclo_a2g
case 1
    for ground_node = N+1:N+n
        maxTemp = abs(RxPowerMatW(ground_node,platcount,ptIdx));
        if maxTemp >= 10^((BckhaulThresh+18)/10)
            pfun = 0.25 * pfun;
        end
    end
otherwise
end

switch pfun_Eval
case 1
    if path_exists(1,platcount)
        pfun = pfun * ((0.5214*gridsize)^2+pos_array(ptIdx,3,platcount)^2);
    end
case 2
    pfun = pfun*2^(max(cont,1)-1);
    if path_exists(1,platcount)
        pfun = pfun * ((0.5214*gridsize)^2+pos_array(ptIdx,3,platcount)^2);
    end
case 3
    if ~path_exists(1,platcount)
        pfun = pfun / ((0.5214*gridsize)^2+pos_array(ptIdx,3,platcount)^2);
    end
    pfun = pfun/2^(N-sum(path_exists(1,1:N)));
case 4
    if path_exists(1,platcount)
        pfun = pfun * ((0.5214*gridsize)^2+pos_array(ptIdx,3,platcount)^2);
    end
end

```

```

        pfun = pfun*2^(sum(path_exists(1,1:N)));
    case 5
        pfun = pfun*2^(max(cont,1)-1);
        if path_exists(1,platcount)
            pfun = 2*pfun;
        end
    case 6
        if cont > 1
            pfun = pfun * ((0.5214*gridsize)^2+pos_array(ptIdx,3,platcount)^2);
        end
        if path_exists(1,platcount)
            pfun = 2*pfun;
        end
    end

    % Move through the other UAVs searching for Rx powers in the
    % arbitrary optimal range. When the Rx power is in the optimal
    % range, pfun is increased, otherwise is penalized (it is not
    % optimal to be so close and it is much worse to be out of range).
    for air_node = 1:N
        minTemp = abs(RxPowerMatW(air_node,platcount,ptIdx));
        if air_node ~= platcount % Conditional to not consider itself
            if minTemp <= 10^((BckhaulThresh+6)/10) && minTemp >=
10^(BckhaulThresh/10)
                g1(platcount, air_node) = plot3(ax,[pos_array(ptIdx,1,platcount)
pos_array(ptIdx,1,air_node)],...
                [pos_array(ptIdx,2,platcount) pos_array(ptIdx,2,air_node)],...
                [pos_array(ptIdx,3,platcount) pos_array(ptIdx,3,air_node)],...
                '--',"Color",[0 1 0],"LineWidth",0.75);
            else
                if minTemp > 10^((BckhaulThresh+18)/10)
                    pfun = 0.25 * pfun;
                    % Create a line plot for the connections between UAVs
                    g1(platcount, air_node) = plot3(ax,[pos_array(ptIdx,1,platcount)
pos_array(ptIdx,1,air_node)],...
                    [pos_array(ptIdx,2,platcount)
pos_array(ptIdx,2,air_node)],...
                    [pos_array(ptIdx,3,platcount)
pos_array(ptIdx,3,air_node)],...
                    '--',"Color",[1 0 0],"LineWidth",0.75);
                end
                if minTemp < 10^(BckhaulThresh/10)
                    pfun = 1 * pfun;
                end
                if minTemp <= 10^((BckhaulThresh+18)/10) && minTemp >
10^(BckhaulThresh+6)/10)
                    pfun = 0.75 * pfun;
                    % Create a line plot for the connections between UAVs
                    g1(platcount, air_node) = plot3(ax,[pos_array(ptIdx,1,platcount)
pos_array(ptIdx,1,air_node)],...
                    [pos_array(ptIdx,2,platcount)
pos_array(ptIdx,2,air_node)],...
                    [pos_array(ptIdx,3,platcount)
pos_array(ptIdx,3,air_node)],...
                    '--',"Color",[1 1 0],"LineWidth",0.75);
                end
            end
        end
    end

```

```

        end
    end
end
minTemp = abs(RxPowerMatW(N+n+1,platcount,ptIdx));
if minTemp <= 10^((VertThresh+6)/10) && minTemp >= 10^((VertThresh)/10)
    % Reward/penalize UAV1 based on its link to ControlSTA
    if platcount == 1 && pen_UAV1_CtrlSta==1
        pfun = pfun * ((0.5214*gridsize)^2+pos_array(ptIdx,3,platcount)^2);
    end
    % Create a line plot for the connections between UAV1 and
    % ControlSTA
    g3(platcount, 1) = plot3(ax,[pos_array(ptIdx,1,platcount)
staPositions(n+1,1)],...
        [pos_array(ptIdx,2,platcount) staPositions(n+1,2)],...
        [pos_array(ptIdx,3,platcount) staPositions(n+1,3)],...
        'h','Color',[0 1 0],"LineWidth",1.75);
else
    if minTemp > 10^((VertThresh+18)/10)
        % Reward/penalize UAV1 based on its link to ControlSTA
        if platcount == 1 && pen_UAV1_CtrlSta==1
            pfun = 0.25 * pfun;
        end
        % Create a line plot for the connections between UAV1
        % and ControlSTA
        g3(platcount, 1) = plot3(ax,[pos_array(ptIdx,1,platcount)
staPositions(n+1,1)],...
            [pos_array(ptIdx,2,platcount) staPositions(n+1,2)],...
            [pos_array(ptIdx,3,platcount) staPositions(n+1,3)],...
            'h',"Color",[1 0.75 0],"LineWidth",1.75);
    end
    if minTemp < 10^((VertThresh)/10)
        % Reward/penalize UAV1 based on its link to ControlSTA
        if platcount == 1 && pen_UAV1_CtrlSta==1
            pfun = 0.25 * pfun;
        end
    end
    if minTemp <= 10^((VertThresh+18)/10) && minTemp > 10^((VertThresh+6)/10)
        % Create a line plot for the connections between UAV1
        % and ControlSTA
        if platcount == 1 && pen_UAV1_CtrlSta==1
            pfun = 0.75 * pfun;
        end
        g3(platcount, 1) = plot3(ax,[pos_array(ptIdx,1,platcount)
staPositions(n+1,1)],...
            [pos_array(ptIdx,2,platcount) staPositions(n+1,2)],...
            [pos_array(ptIdx,3,platcount) staPositions(n+1,3)],...
            'h',"Color",[1 1 0],"LineWidth",1.75);
    end
end
end
end

```

## APPENDIX B: Implementation of Dijkstra's Algorithm in MATLAB

```
function [Cost, Route] = Dijkstras( Graph, SourceNode, TerminalNode )
%Dijkstras.m Given a graph with distances from node to node calculates the
%optimal route from the Source Node to the Terminal Node as defined by the
%inputs.
% Check for valid parameters
if size(Graph,1) ~= size(Graph,2)
    fprintf('The Graph must be a square Matrix\n');
    return;
elseif min(min(Graph)) < 0
    fprintf('Dijkstras algorithm cannot handle negative costs.\n');
    fprintf('Please use Bellman-Ford or another alternative instead\n');
    return;
elseif SourceNode < 1 && (rem(SourceNode,1)==0) && (isreal(SourceNode)) &&
(SourceNode <= size(Graph,1))
    fprintf('The source node must be an integer within [1, sizeofGraph]\n');
    return;
elseif TerminalNode < 1 && (rem(TerminalNode,1)==0) && isreal(TerminalNode) &&
(TerminalNode <= size(Graph,1))
    fprintf('The terminal node must be an integer within [1, sizeofGraph]\n');
    return;
end

% Special Case so no need to waste time doing initializations
if SourceNode == TerminalNode
    Cost = Graph(SourceNode, TerminalNode);
    Route = SourceNode;
    return;
end

% Set up a cell structure so that I can store the optimal path from source
% node to each node in this structure. This structure stores the
% antecedents so for instance if there is a path to B through A-->C-->D-->B
% you will see [A,C,D] in cell{B} (as well as a bunch of filler 0's after
% that)
PathToNode = cell(size(Graph,1),1);

% Initialize all Node costs to infinity except for the source node
NodeCost = Inf.*ones(1,size(Graph,1));
NodeCost(SourceNode) = 0;

% Initialize the Current Node to be the Source Node
CurrentNode = SourceNode;

% Initialize the set of Visited and Unvisited Nodes
VisitedNodes = SourceNode;
UnvisitedNodes = 1:size(Graph,2);
UnvisitedNodes = UnvisitedNodes(UnvisitedNodes ~= VisitedNodes);

while (CurrentNode ~= TerminalNode)
    % Extract the Costs/Path Lengths to each node from the current node
    CostVector = Graph(CurrentNode, :);
    % Only look at valid neighbors ie. those nodes which are unvisited
    UnvisitedNeighborsCostVector = CostVector(UnvisitedNodes);
```

```

% Extract the cost to get to the Current Node
CurrentNodeCost = NodeCost(CurrentNode);
% Extract the path to the current node
PathToCurrentNode = PathToNode{CurrentNode};
% Iterate through the Unvisited Neighbors assigning them a new tentative cost
for i = 1:length(UnvisitedNeighborsCostVector)
    if UnvisitedNeighborsCostVector(i) ~= Inf % Only Check for update if non-
infinite
        tempCost = CurrentNodeCost + UnvisitedNeighborsCostVector(i); % The
tentative cost to get to the neighbor through the current node
        % Compare the tentative cost to the currently assigned cost and
        % assign the minimum
        if tempCost < NodeCost(UnvisitedNodes(i))
            NewPathToNeighbor = [PathToCurrentNode(PathToCurrentNode~=0)
CurrentNode]; % The new path to get to the neighbor
            NewPath = [NewPathToNeighbor zeros(1,size(Graph,1)-
size(NewPathToNeighbor,2))];
            PathToNode{UnvisitedNodes(i)}(:) = NewPath;
            NodeCost(UnvisitedNodes(i)) = tempCost;
        end
    end
end
% Search for the smallest cost remaining that is in the unvisited set
RemainingCosts = NodeCost(UnvisitedNodes);
[MIN, MIN_IND] = min(RemainingCosts);

% If the smallest remaining cost amongst the unvisited set of nodes is
% infinite then there is no valid path from the source node to the
% terminal node.
if MIN == Inf
    Cost = Inf;
    Route = [];
    return;
end

% Update the Visited and Unvisited Nodes
VisitedNodes = [VisitedNodes CurrentNode];
CurrentNode = UnvisitedNodes(MIN_IND);
UnvisitedNodes = UnvisitedNodes(UnvisitedNodes~=CurrentNode);
end

Route = PathToNode{TerminalNode};
Route = Route(Route~=0);
Route = [Route TerminalNode];
Cost = NodeCost(TerminalNode);
end

```

## APPENDIX C: Modified MATLAB Functions to Load Configuration Parameters

```
function [nodeConfigs, trafficConfigs] = hLoadConfigurationFull_Int_Traff_6(N, n,
apPositions, staPositions, VertThresh)
%loadConfiguration Returns the node and traffic configuration

numRooms = 1;
numAPs = N;
numSTAs = n;
%numAPPerRoom = numAPs/numRooms; % One AP in each room
numSTAPerRoom = numSTAs/numRooms;
numNodes = numAPs + numSTAs;

% Get the node IDs and positions for all the nodes
[nodeIDs, positions] = hGetIDsAndPositions(numAPs, numSTAs, apPositions,
staPositions);

% Load the application traffic configuration for WLAN nodes
s = load('wlanTrafficConfig.mat', 'wlanTrafficConfig');

% Configure application traffic such that each AP has traffic for all STAs
% present in same room.
traffSize = n;
trafficConfigs = repmat(s.wlanTrafficConfig, 1, traffSize);
% ControlSTA (Grndnd to ControlSTA)
for rooomIdx = 1:n
    trafficConfigs(rooomIdx).SourceNode = ['Node' num2str(rooomIdx)];
    trafficConfigs(rooomIdx).DestinationNode = ['ControlSTA' '1'];
    trafficConfigs(rooomIdx).DataRateKbps = 100000;
end

% Load the node configuration structure and initialize for all the nodes
s = load('wlanNodeConfig.mat', 'wlanNodeConfig');
nodeConfigs = repmat(s.wlanNodeConfig, 1, numNodes+1);

% Customize configuration for nodes
% Set node positions in each node configuration
for nodeIdx = 1:numAPs
    nodeConfigs(nodeIdx).NodeName = ['UAV' num2str(nodeIdx)];
    nodeConfigs(nodeIdx).NodePosition = apPositions(nodeIdx,:);
    nodeConfigs(nodeIdx).IsMeshNode = 1;
    nodeConfigs(nodeIdx).EDThreshold = VertThresh;
end

for nodeIdx = numAPs+1:numNodes
    nodeConfigs(nodeIdx).NodeName = ['Node' num2str(nodeIdx-numAPs)];
    nodeConfigs(nodeIdx).NodePosition = staPositions(nodeIdx-numAPs,:);
    nodeConfigs(nodeIdx).IsMeshNode = 1;
    nodeConfigs(nodeIdx).EDThreshold = VertThresh;
end

nodeConfigs(numNodes+1).NodeName = ['ControlSTA' '1'];
nodeConfigs(numNodes+1).NodePosition = staPositions(numSTAs+1,:);
nodeConfigs(numNodes+1).IsMeshNode = 1;
nodeConfigs(numNodes+1).EDThreshold = VertThresh;
```

```

end

function [nodeIDs, positions] = hGetIDsAndPositions(N, n, apPositions, staPositions)
%hGetIDsAndPositions Returns the IDs and positions of nodes in the network

numRooms = 1;
numAPs = N;
numSTAs = n;
%numAPPerRoom = numAPs/numRooms; % N APs in each room
numSTAPerRoom = numSTAs/numRooms;
numNodes = numAPs + numSTAs;

apNodeIDs = (1:numAPs)';
staNodeIDs = (numAPs+1:numNodes);

nodeIDs = zeros(numAPs, numSTAPerRoom+1);

positions = cell(numAPs, numSTAPerRoom+1);

% Assign IDs and positions to each node
nodeIDs(:, 1) = apNodeIDs;
for roomIdx = 1:numAPs
    positions{roomIdx, 1} = apPositions(roomIdx, :);

    for staIdx = 1:numSTAPerRoom
        nodeIDs(roomIdx, staIdx+1) = staNodeIDs(staIdx);
        positions{roomIdx, staIdx+1} = staPositions(staIdx, :);
    end
end
end

function [nodeConfigs, trafficConfigs] = hLoadConfigurationFull_Int_Traff_7(N, n,
apPositions, staPositions, VertThresh)
%loadConfiguration Returns the node and traffic configuration

numRooms = 1;
numAPs = N;
numSTAs = n;
%numAPPerRoom = numAPs/numRooms; % One AP in each room
numSTAPerRoom = numSTAs/numRooms;
numNodes = numAPs + numSTAs;

% Get the node IDs and positions for all the nodes
[nodeIDs, positions] = hGetIDsAndPositions(numAPs, numSTAs, apPositions,
staPositions);

% Load the application traffic configuration for WLAN nodes
s = load('wlanTrafficConfig.mat', 'wlanTrafficConfig');

% Configure application traffic such that each AP has traffic for all STAs
% present in same room.
traffSize = n;
trafficConfigs = repmat(s.wlanTrafficConfig, 1, traffSize);

```

```

% ControlSTA (Grndnd to ControlSTA)
for roomIdx = 1:n
    trafficConfigs(roomIdx).SourceNode = ['Node' num2str(roomIdx)];
    trafficConfigs(roomIdx).DestinationNode = ['ControlSTA' '1'];
    trafficConfigs(roomIdx).PacketSize = 250;
    trafficConfigs(roomIdx).DataRateKbps = 62.5;
end

% Load the node configuration structure and initialize for all the nodes
s = load('wlanNodeConfig.mat', 'wlanNodeConfig');
nodeConfigs = repmat(s.wlanNodeConfig, 1, numNodes+1);

% Customize configuration for nodes
% Set node positions in each node configuration
for nodeIdx = 1:numAPs
    nodeConfigs(nodeIdx).NodeName = ['UAV' num2str(nodeIdx)];
    nodeConfigs(nodeIdx).NodePosition = apPositions(nodeIdx,:);
    nodeConfigs(nodeIdx).IsMeshNode = 1;
    nodeConfigs(nodeIdx).EDThreshold = VertThresh;
end

for nodeIdx = numAPs+1:numNodes
    nodeConfigs(nodeIdx).NodeName = ['Node' num2str(nodeIdx-numAPs)];
    nodeConfigs(nodeIdx).NodePosition = staPositions(nodeIdx-numAPs,:);
    nodeConfigs(nodeIdx).IsMeshNode = 1;
    nodeConfigs(nodeIdx).EDThreshold = VertThresh;
end

nodeConfigs(numNodes+1).NodeName = ['ControlSTA' '1'];
nodeConfigs(numNodes+1).NodePosition = staPositions(numSTAs+1,:);
nodeConfigs(numNodes+1).IsMeshNode = 1;
nodeConfigs(numNodes+1).EDThreshold = VertThresh;
end

function [nodeIDs, positions] = hGetIDsAndPositions(N, n, apPositions, staPositions)
%hGetIDsAndPositions Returns the IDs and positions of nodes in the network

numRooms = 1;
numAPs = N;
numSTAs = n;
%numAPPerRoom = numAPs/numRooms; % N APs in each room
numSTAPerRoom = numSTAs/numRooms;
numNodes = numAPs + numSTAs;

apNodeIDs = (1:numAPs)';
staNodeIDs = (numAPs+1:numNodes);

nodeIDs = zeros(numAPs, numSTAPerRoom+1);

positions = cell(numAPs, numSTAPerRoom+1);

% Assign IDs and positions to each node
nodeIDs(:, 1) = apNodeIDs;
for roomIdx = 1:numAPs
    positions{roomIdx, 1} = apPositions(roomIdx, :);

```



```
    for staIdx = 1:numSTAPerRoom
        nodeIDs(roomIdx, staIdx+1) = staNodeIDs(staIdx);
        positions{roomIdx, staIdx+1} = staPositions(staIdx, :);
    end
end
end
```

## APPENDIX D: Function Used to Obtain the Path Loss Table

```
function [pl,pathlossFn] = hCreatePathlossTableDP(txs,rxs,propModel,varargin)
%hCreatePathlossTable Create path loss table from transmitter and receiver sites

[numFreqs,numNodes] = size(txs);
assert(isequal(size(txs),size(rxs)))

% Use first column to get frequencies used
uniqueFreqs = [txs(:,1).TransmitterFrequency];

% Rows are transmitters, columns are receivers
pl = zeros(numNodes,numNodes,numFreqs);
for i = 1:numFreqs
    plf = pathloss(propModel,rxs(i,:),txs(i,:));
    % Make pathloss for links reciprocal - shadow fading may cause them not
    % to be when generated with pathloss
    if isempty(varargin)
        pl(:, :, i) = triu(plf) + triu(plf,1)';
    else
        N = size(plf,1);
        for countr = 1:(N)
            for countc =1:(N)
                if plf{countr,countc} ~= 0
                    pl(countr,countc) = [plf{countr,countc}];
                else
                    pl(countr,countc) = 0;
                end
            end
        end
    end
end

% Handle to lookup table anonymous function
pathlossFn = @(txIdx,rxIdx,freq) pl(txIdx,rxIdx,freq==uniqueFreqs);

end
```

## APPENDIX E: Function to Implement the Free Space Propagation Model

```
classdef hFreeSpacePathLoss < rfprop.PropagationModel

    properties
        Triangulation;

        TriangulationUnit (1,1) double {mustBeFinite, mustBeReal, mustBeNonnegative,
mustBeNonzero, mustBeNonsparse} = 1;

        FacesPerWall (1,1) double {mustBeFinite, mustBeReal, mustBeNonnegative,
mustBeNonzero, mustBeNonsparse} = 2;

        ShadowSigma (1,1) double {mustBeFinite, mustBeReal, mustBeNonnegative,
mustBeNonsparse} = 5;

        WallZThreshold (1,1) double {mustBeFinite, mustBeReal, mustBeNonnegative,
mustBeNonzero, mustBeNonsparse} = 0.1;

        FloorThicknessThreshold (1,1) double {mustBeFinite, mustBeReal,
mustBeNonnegative, mustBeNonzero, mustBeNonsparse} = 0.6;
    end

    methods
        function [numFloors,numWalls,distance] = linkInfo(plm,txs,rxs)

            % Determine number of floors and walls penetrated for each link
            [~,numFloors,numWalls,distance] = wlanresidentialpnl(plm,txs,rxs);
        end

        function visualizeLinkInfo(plm,tx,rx)
            visualize = true;
            hTGaxIndoorLinkInfo(plm.Triangulation, tx, rx, plm.TriangulationUnit,
visualize, ...
                'FacesPerWall', plm.FacesPerWall, ...
                'WallZThreshold', plm.WallZThreshold, ...
                'FloorThicknessThreshold', plm.FloorThicknessThreshold);
        end
    end

    methods(Access = protected)
        function pl = pathlossOverDistance(pm, rxs, tx, d, ~)
            % Scale distance into meters
            d = d*pm.TriangulationUnit;

            % Path loss
            pl = wlanresidentialpl(pm,d,tx.TransmitterFrequency);

            % Penetration loss
            pnl = wlanresidentialpnl(pm,tx,rxs);

            % Large-scale shadow fading
            sf = pm.ShadowSigma*randn(size(pl));

            pl = pl + pnl + sf;
        end
    end
end
```

```

end
end

methods (Access = protected)

function L = wlanresidentialpl(~,R,freq)
    %wlanresidentialpl    wlanresidential path loss
    %    L = wlanresidential() returns the WLAN Residential scenario path loss
in dB.
    %
    %    Note that the best case is lossless, so the loss is always greater
than
    %    or equal to 0 dB.

    R = max(R,1); % minimum distance is 1 meter
    L = 40.05+20*log10(freq/2.4e9) + 20*log10(R);
end

function [L,numFloors,numWalls,distance] = wlanresidentialpnl(plm,txs,rxs)
    %wlanresidentialpnl    wlanresidential penetration loss
    %    L = wlanresidentialpnl() returns the WLAN Residential scenario path
loss in dB.
    %
    %    Note that the best case is lossless, so the loss is always greater
than
    %    or equal to 0 dB.

    % Determine number of floors and walls penetrated for each link
    [numFloors,numWalls,distance] = hTGaxIndoorLinkInfo(plm.Triangulation,
txs, rxs, plm.TriangulationUnit, ...
    'FacesPerWall', plm.FacesPerWall, ...
    'WallZThreshold', plm.WallZThreshold, ...
    'FloorThicknessThreshold', plm.FloorThicknessThreshold);

    % penetration loss
    L = 18.3*numFloors.^((numFloors+2)./(numFloors+1) -0.46) + 5*numWalls;
end

end

end

```

## APPENDIX F: Performance Evaluation of the Different Mobility Algorithms (Tabulated Results)

### Ray tracing

#### FIXED

| No. UAVs | GN | ITERATIONS 1s | COVERAGE | EFFICIENCY | PATH | COMM EFF | nf | ns  | NF | UAVS WITH PATH TO CS | UAVS CONNECTED TO CS | NTX nodes | NTX nodes reach CS |
|----------|----|---------------|----------|------------|------|----------|----|-----|----|----------------------|----------------------|-----------|--------------------|
| 1        | 10 | 48            | 40%      | 40%        | 0%   | 0%       | 4  | 4   | 1  | 0                    | 0                    | 0         | 0                  |
| 2        | 10 | 61            | 70%      | 54%        | 0%   | 0%       | 7  | 9   | 2  | 0                    | 0                    | 0         | 0                  |
| 3        | 10 | 65            | 70%      | 54%        | 0%   | 0%       | 7  | 9   | 3  | 0                    | 0                    | 0         | 0                  |
| 4        | 10 | 71            | 100%     | 77%        | 20%  | 15%      | 10 | 13  | 4  | 1                    | 1                    | 2         | 0                  |
| 5        | 10 | 75            | 80%      | 49%        | 30%  | 15%      | 8  | 13  | 5  | 3                    | 1                    | 3         | 1                  |
| 6        | 10 | 75            | 100%     | 56%        | 30%  | 17%      | 10 | 18  | 6  | 3                    | 1                    | 3         | 1                  |
| 7        | 10 | 77            | 100%     | 50%        | 30%  | 15%      | 10 | 20  | 7  | 4                    | 1                    | 3         | 1                  |
| 8        | 10 | 77            | 100%     | 40%        | 30%  | 12%      | 10 | 25  | 8  | 4                    | 1                    | 3         | 2                  |
| 9        | 10 | 79            | 100%     | 40%        | 100% | 40%      | 10 | 25  | 9  | 9                    | 1                    | 10        | 3                  |
| 10       | 20 | 81            | 95%      | 27%        | 95%  | 26%      | 19 | 66  | 10 | 10                   | 1                    | 19        | 4                  |
| 15       | 20 | 82            | 100%     | 22%        | 100% | 22%      | 20 | 92  | 15 | 15                   | 2                    | 20        | 2                  |
| 16       | 20 | 83            | 100%     | 19%        | 100% | 19%      | 20 | 107 | 16 | 16                   | 1                    | 20        | 1                  |
| 25       | 30 | 85            | 100%     | 13%        | 100% | 13%      | 30 | 229 | 25 | 25                   | 3                    | 30        | 4                  |

#### HYBRID

| No. UAVs | GN | ITERATIONS 1s | COVERAGE | EFFICIENCY | PATH | COMM EFF | nf | ns | NF | UAVS WITH PATH TO CS | UAVS CONNECTED TO CS | NTX nodes | NTX nodes reach CS |
|----------|----|---------------|----------|------------|------|----------|----|----|----|----------------------|----------------------|-----------|--------------------|
| 1        | 10 | 81            | 20%      | 20%        | 20%  | 4%       | 2  | 2  | 1  | 1                    | 1                    | 2         | 1                  |
| 2        | 10 | 82            | 30%      | 18%        | 30%  | 5%       | 3  | 5  | 2  | 2                    | 1                    | 3         | 1                  |
| 3        | 10 | 87            | 30%      | 15%        | 30%  | 5%       | 3  | 6  | 3  | 3                    | 1                    | 3         | 1                  |

|    |    |     |      |     |      |     |    |     |    |    |   |    |   |
|----|----|-----|------|-----|------|-----|----|-----|----|----|---|----|---|
| 4  | 10 | 180 | 80%  | 40% | 80%  | 32% | 8  | 16  | 4  | 4  | 1 | 8  | 1 |
| 5  | 10 | 101 | 80%  | 40% | 80%  | 32% | 8  | 16  | 5  | 5  | 1 | 8  | 1 |
| 6  | 10 | 101 | 100% | 45% | 30%  | 14% | 10 | 22  | 6  | 3  | 1 | 3  | 1 |
| 7  | 10 | 128 | 80%  | 29% | 80%  | 23% | 8  | 22  | 7  | 7  | 1 | 8  | 1 |
| 8  | 10 | 166 | 90%  | 29% | 90%  | 26% | 9  | 28  | 8  | 8  | 1 | 9  | 1 |
| 9  | 10 | 180 | 100% | 36% | 100% | 36% | 10 | 28  | 9  | 9  | 2 | 10 | 2 |
| 10 | 20 | 180 | 80%  | 17% | 80%  | 13% | 16 | 77  | 10 | 10 | 1 | 16 | 3 |
| 15 | 20 | 180 | 85%  | 12% | 85%  | 10% | 17 | 117 | 15 | 15 | 2 | 17 | 4 |
| 16 | 20 | 180 | 100% | 16% | 100% | 16% | 20 | 123 | 16 | 16 | 1 | 20 | 6 |
| 25 | 30 | 180 | 100% | 10% | 100% | 10% | 30 | 290 | 25 | 25 | 3 | 30 | 3 |

PSO-  
only

| No. UAVs | GN | ITERATIONS 1s | COVERAGE | EFFICIENCY | PATH | COMM EFF | nf | ns  | NF | UAVS WITH PATH TO CS | UAVS CONNECTED TO CS | NTX nodes | NTX nodes reach CS |
|----------|----|---------------|----------|------------|------|----------|----|-----|----|----------------------|----------------------|-----------|--------------------|
| 1        | 10 | 21            | 20%      | 20%        | 20%  | 4%       | 2  | 2   | 1  | 1                    | 1                    | 2         | 1                  |
| 2        | 10 | 137           | 20%      | 13%        | 20%  | 3%       | 2  | 3   | 2  | 2                    | 2                    | 2         | 1                  |
| 3        | 10 | 180           | 30%      | 18%        | 30%  | 5%       | 3  | 5   | 3  | 3                    | 2                    | 3         | 1                  |
| 4        | 10 | 180           | 40%      | 23%        | 10%  | 2%       | 4  | 7   | 4  | 2                    | 2                    | 1         | 1                  |
| 5        | 10 | 177           | 60%      | 33%        | 60%  | 20%      | 6  | 11  | 5  | 5                    | 2                    | 6         | 4                  |
| 6        | 10 | 175           | 70%      | 20%        | 70%  | 14%      | 7  | 24  | 6  | 6                    | 1                    | 7         | 1                  |
| 7        | 10 | 173           | 80%      | 28%        | 20%  | 6%       | 8  | 23  | 7  | 3                    | 2                    | 2         | 1                  |
| 8        | 10 | 180           | 90%      | 23%        | 10%  | 2%       | 9  | 35  | 8  | 1                    | 1                    | 1         | 1                  |
| 9        | 10 | 180           | 90%      | 30%        | 60%  | 18%      | 9  | 27  | 9  | 8                    | 3                    | 6         | 1                  |
| 10       | 20 | 180           | 95%      | 24%        | 75%  | 18%      | 19 | 76  | 10 | 9                    | 3                    | 15        | 2                  |
| 15       | 20 | 180           | 80%      | 10%        | 80%  | 8%       | 16 | 134 | 15 | 15                   | 2                    | 16        | 2                  |
| 16       | 20 | 180           | 90%      | 12%        | 90%  | 11%      | 18 | 133 | 16 | 16                   | 3                    | 18        | 1                  |
| 25       | 30 | 180           | 87%      | 9%         | 87%  | 7%       | 26 | 317 | 30 | 25                   | 6                    | 26        | 5                  |

## Log-normal

### FIXED

| No. UAVs | GN | ITERATIONS 1s | COVERAGE | EFFICIENCY | PATH | COMM EFF | nf | ns | NF | UAVS WITH PATH TO CS | UAVS CONNECTED TO CS | NTX nodes | NTX nodes reach CS |
|----------|----|---------------|----------|------------|------|----------|----|----|----|----------------------|----------------------|-----------|--------------------|
| 1        | 10 | 48            | 20%      | 20%        | 0%   | 0%       | 2  | 2  | 1  | 0                    | 0                    | 0         | 0                  |
| 2        | 10 | 61            | 30%      | 30%        | 0%   | 0%       | 3  | 3  | 2  | 0                    | 0                    | 0         | 0                  |
| 3        | 10 | 65            | 50%      | 50%        | 0%   | 0%       | 5  | 5  | 3  | 0                    | 0                    | 0         | 0                  |
| 4        | 10 | 71            | 80%      | 64%        | 0%   | 0%       | 8  | 10 | 4  | 0                    | 0                    | 0         | 0                  |
| 5        | 10 | 75            | 70%      | 61%        | 0%   | 0%       | 7  | 8  | 5  | 0                    | 0                    | 0         | 0                  |
| 6        | 10 | 75            | 100%     | 83%        | 0%   | 0%       | 10 | 12 | 6  | 0                    | 0                    | 0         | 0                  |
| 7        | 10 | 77            | 80%      | 49%        | 30%  | 15%      | 8  | 13 | 7  | 4                    | 1                    | 3         | 1                  |
| 8        | 10 | 77            | 100%     | 67%        | 30%  | 20%      | 10 | 15 | 8  | 4                    | 1                    | 3         | 1                  |
| 9        | 10 | 79            | 100%     | 59%        | 10%  | 6%       | 10 | 17 | 9  | 1                    | 1                    | 1         | 0                  |
| 10       | 20 | 81            | 95%      | 48%        | 40%  | 19%      | 19 | 38 | 10 | 4                    | 1                    | 8         | 3                  |
| 15       | 20 | 82            | 100%     | 33%        | 40%  | 13%      | 20 | 61 | 15 | 4                    | 1                    | 8         | 2                  |
| 16       | 20 | 83            | 100%     | 34%        | 100% | 34%      | 20 | 58 | 16 | 16                   | 1                    | 20        | 4                  |

### HYBRID

| No. UAVs | GN | ITERATIONS 1s | COVERAGE | EFFICIENCY | PATH | COMM EFF | nf | ns | NF | UAVS WITH PATH TO CS | UAVS CONNECTED TO CS | NTX nodes | NTX nodes reach CS |
|----------|----|---------------|----------|------------|------|----------|----|----|----|----------------------|----------------------|-----------|--------------------|
| 1        | 10 | 75            | 10%      | 10%        | 10%  | 1%       | 1  | 1  | 1  | 1                    | 1                    | 1         | 0                  |
| 2        | 10 | 88            | 20%      | 13%        | 20%  | 3%       | 2  | 3  | 2  | 2                    | 1                    | 2         | 1                  |
| 3        | 10 | 88            | 30%      | 15%        | 30%  | 5%       | 3  | 6  | 3  | 3                    | 1                    | 3         | 1                  |
| 4        | 10 | 100           | 60%      | 45%        | 10%  | 5%       | 6  | 8  | 4  | 2                    | 2                    | 1         | 1                  |
| 5        | 10 | 101           | 80%      | 53%        | 30%  | 16%      | 8  | 12 | 5  | 3                    | 1                    | 3         | 1                  |

|    |    |     |      |     |      |     |    |     |    |    |   |    |   |
|----|----|-----|------|-----|------|-----|----|-----|----|----|---|----|---|
| 6  | 10 | 103 | 80%  | 43% | 20%  | 9%  | 8  | 15  | 6  | 3  | 1 | 2  | 2 |
| 7  | 10 | 180 | 80%  | 32% | 80%  | 26% | 8  | 20  | 7  | 7  | 1 | 8  | 2 |
| 8  | 10 | 180 | 90%  | 31% | 90%  | 28% | 9  | 26  | 8  | 6  | 1 | 9  | 1 |
| 9  | 10 | 102 | 100% | 45% | 100% | 45% | 10 | 22  | 9  | 9  | 2 | 10 | 3 |
| 10 | 20 | 180 | 80%  | 17% | 80%  | 14% | 16 | 74  | 10 | 10 | 1 | 16 | 2 |
| 15 | 20 | 180 | 100% | 19% | 100% | 19% | 20 | 107 | 15 | 15 | 4 | 20 | 2 |
| 16 | 20 | 107 | 100% | 19% | 100% | 19% | 20 | 108 | 16 | 16 | 1 | 20 | 5 |

**PSO-  
only**

| No. UAVs | GN | ITERATIONS 1s | COVER AGE | EFICIENCY | PATH | COMM EFF | nf | ns  | NF | UAVS WITH PATH TO CS | UAVS CONNECTED TO CS | NTX nodes | NTX nodes reach CS |
|----------|----|---------------|-----------|-----------|------|----------|----|-----|----|----------------------|----------------------|-----------|--------------------|
| 1        | 10 | 23            | 20%       | 20%       | 20%  | 4%       | 2  | 2   | 1  | 1                    | 1                    | 2         | 1                  |
| 2        | 10 | 180           | 10%       | 5%        | 10%  | 1%       | 1  | 2   | 2  | 2                    | 2                    | 1         | 1                  |
| 3        | 10 | 151           | 40%       | 27%       | 40%  | 11%      | 4  | 6   | 3  | 3                    | 1                    | 4         | 1                  |
| 4        | 10 | 180           | 50%       | 36%       | 10%  | 4%       | 5  | 7   | 4  | 2                    | 2                    | 1         | 1                  |
| 5        | 10 | 177           | 30%       | 13%       | 30%  | 4%       | 3  | 7   | 5  | 5                    | 2                    | 3         | 2                  |
| 6        | 10 | 177           | 40%       | 16%       | 40%  | 6%       | 4  | 10  | 6  | 6                    | 3                    | 4         | 1                  |
| 7        | 10 | 176           | 60%       | 26%       | 60%  | 15%      | 6  | 14  | 7  | 7                    | 2                    | 6         | 1                  |
| 8        | 10 | 180           | 80%       | 22%       | 80%  | 18%      | 8  | 29  | 8  | 8                    | 2                    | 8         | 1                  |
| 9        | 10 | 180           | 80%       | 34%       | 80%  | 27%      | 8  | 19  | 9  | 9                    | 4                    | 8         | 0                  |
| 10       | 20 | 174           | 70%       | 12%       | 70%  | 9%       | 14 | 79  | 10 | 10                   | 2                    | 14        | 1                  |
| 15       | 20 | 180           | 70%       | 8%        | 70%  | 5%       | 14 | 125 | 15 | 15                   | 3                    | 14        | 1                  |
| 16       | 20 | 180           | 80%       | 11%       | 80%  | 9%       | 16 | 118 | 16 | 16                   | 2                    | 16        | 3                  |



## Log-normal + LoRa

### FIXED

| No. UAVs | GN | ITERATIONS 1s | COVERAGE | EFFICIENCY | PATH | COMM EFF | nf | ns | NF | UAVS WITH PATH TO CS | UAVS CONNECTED TO CS | NTX nodes | NTX nodes reach CS |
|----------|----|---------------|----------|------------|------|----------|----|----|----|----------------------|----------------------|-----------|--------------------|
| 1        | 10 | 48            | 20%      | 20%        | 0%   | 0%       | 2  | 2  | 1  | 0                    | 0                    | 0         | 0                  |
| 2        | 10 | 61            | 30%      | 30%        | 0%   | 0%       | 3  | 3  | 2  | 0                    | 0                    | 0         | 0                  |
| 3        | 10 | 65            | 50%      | 50%        | 0%   | 0%       | 5  | 5  | 3  | 0                    | 0                    | 0         | 0                  |
| 4        | 10 | 71            | 80%      | 64%        | 0%   | 0%       | 8  | 10 | 4  | 0                    | 0                    | 0         | 0                  |
| 5        | 10 | 75            | 70%      | 61%        | 0%   | 0%       | 7  | 8  | 5  | 0                    | 0                    | 0         | 0                  |
| 6        | 10 | 75            | 100%     | 83%        | 0%   | 0%       | 10 | 12 | 6  | 0                    | 0                    | 0         | 0                  |
| 7        | 10 | 77            | 80%      | 49%        | 80%  | 39%      | 8  | 13 | 7  | 7                    | 1                    | 8         | 3                  |
| 8        | 10 | 77            | 100%     | 67%        | 100% | 67%      | 10 | 15 | 8  | 8                    | 1                    | 10        | 2                  |
| 9        | 10 | 79            | 100%     | 59%        | 100% | 59%      | 10 | 17 | 9  | 9                    | 1                    | 10        | 1                  |
| 10       | 20 | 81            | 95%      | 48%        | 95%  | 45%      | 19 | 38 | 10 | 10                   | 1                    | 19        | 3                  |
| 15       | 20 | 82            | 100%     | 33%        | 100% | 33%      | 20 | 61 | 15 | 15                   | 1                    | 20        | 5                  |
| 16       | 20 | 83            | 100%     | 34%        | 100% | 34%      | 20 | 58 | 16 | 16                   | 1                    | 20        | 5                  |

### HYBRID

| No. UAVs | GN | ITERATIONS 1s | COVERAGE | EFFICIENCY | PATH | COMM EFF | nf | ns | NF | UAVS WITH PATH TO CS | UAVS CONNECTED TO CS | NTX nodes | NTX nodes reach CS |
|----------|----|---------------|----------|------------|------|----------|----|----|----|----------------------|----------------------|-----------|--------------------|
| 1        | 10 | 73            | 20%      | 20%        | 20%  | 4%       | 2  | 2  | 1  | 1                    | 1                    | 2         | 1                  |
| 2        | 10 | 79            | 30%      | 30%        | 30%  | 9%       | 3  | 3  | 2  | 2                    | 1                    | 3         | 2                  |
| 3        | 10 | 91            | 40%      | 27%        | 40%  | 11%      | 4  | 6  | 3  | 3                    | 2                    | 4         | 1                  |
| 4        | 10 | 99            | 80%      | 64%        | 80%  | 51%      | 8  | 10 | 4  | 4                    | 2                    | 8         | 4                  |
| 5        | 10 | 101           | 80%      | 53%        | 80%  | 43%      | 8  | 12 | 5  | 5                    | 1                    | 8         | 4                  |

|    |    |     |      |     |      |     |    |     |    |    |   |    |   |
|----|----|-----|------|-----|------|-----|----|-----|----|----|---|----|---|
| 6  | 10 | 101 | 100% | 56% | 100% | 56% | 10 | 18  | 6  | 6  | 1 | 10 | 3 |
| 7  | 10 | 180 | 80%  | 43% | 80%  | 34% | 8  | 15  | 7  | 7  | 2 | 8  | 3 |
| 8  | 10 | 180 | 90%  | 39% | 90%  | 35% | 9  | 21  | 8  | 8  | 2 | 9  | 3 |
| 9  | 10 | 139 | 90%  | 30% | 90%  | 27% | 9  | 27  | 9  | 9  | 2 | 9  | 2 |
| 10 | 20 | 180 | 75%  | 18% | 75%  | 14% | 15 | 61  | 10 | 10 | 4 | 15 | 6 |
| 15 | 20 | 180 | 90%  | 12% | 90%  | 11% | 18 | 96  | 11 | 15 | 4 | 18 | 4 |
| 16 | 20 | 129 | 75%  | 8%  | 75%  | 6%  | 15 | 106 | 12 | 16 | 5 | 15 | 5 |

**PSO-  
only**

| No. UAVs | GN | ITERATIONS 1s | COVER AGE | EFICIENCY | PATH | COMM EFF | nf | ns  | NF | UAVS WITH PATH TO CS | UAVS CONNECTED TO CS | NTX nodes | NTX nodes reach CS |
|----------|----|---------------|-----------|-----------|------|----------|----|-----|----|----------------------|----------------------|-----------|--------------------|
| 1        | 10 | 22            | 20%       | 20%       | 20%  | 4%       | 2  | 2   | 1  | 1                    | 1                    | 2         | 1                  |
| 2        | 10 | 180           | 10%       | 5%        | 10%  | 1%       | 1  | 2   | 2  | 2                    | 1                    | 1         | 1                  |
| 3        | 10 | 180           | 10%       | 3%        | 10%  | 0%       | 1  | 3   | 3  | 3                    | 3                    | 1         | 1                  |
| 4        | 10 | 112           | 70%       | 49%       | 70%  | 34%      | 7  | 10  | 4  | 4                    | 2                    | 7         | 3                  |
| 5        | 10 | 174           | 20%       | 7%        | 20%  | 1%       | 2  | 6   | 5  | 5                    | 3                    | 2         | 2                  |
| 6        | 10 | 174           | 50%       | 25%       | 50%  | 13%      | 5  | 10  | 6  | 6                    | 5                    | 5         | 1                  |
| 7        | 10 | 169           | 60%       | 19%       | 60%  | 12%      | 6  | 16  | 6  | 6                    | 2                    | 6         | 1                  |
| 8        | 10 | 180           | 90%       | 31%       | 90%  | 28%      | 9  | 26  | 8  | 8                    | 3                    | 9         | 2                  |
| 9        | 10 | 125           | 60%       | 21%       | 60%  | 13%      | 6  | 17  | 9  | 9                    | 4                    | 6         | 4                  |
| 10       | 20 | 166           | 65%       | 14%       | 65%  | 9%       | 13 | 59  | 10 | 10                   | 7                    | 13        | 3                  |
| 15       | 20 | 178           | 75%       | 10%       | 75%  | 7%       | 15 | 118 | 15 | 15                   | 4                    | 15        | 4                  |
| 16       | 20 | 71            | 80%       | 11%       | 80%  | 8%       | 16 | 121 | 16 | 16                   | 2                    | 16        | 2                  |