



YAŞAR UNIVERSITY

GRADUATE SCHOOL

MASTER OF SCIENCE THESIS

**TASK AND PATH PLANNING IN
MULTIROBOT SYSTEMS**

AHMET ENES OK

THESIS ADVISOR: ASSIST.PROF.(PHD) DENİZ ÖZSOYELLER

MSC IN ARTIFICIAL INTELLIGENCE ENGINEERING

BORNOVA / İZMİR
AUGUST 2025

JURY APPROVAL PAGE

We certify that, as the jury, we have read this thesis and that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master in Science.

Jury Members:

Signature:

Assist.Prof.(PhD) Deniz Özsoyeller
Yasar University

.....

Assist.Prof.(PhD) Kazım Erdoğan
Yasar University

.....

Assist.Prof.(PhD) İlker Korkmaz
İzmir University of Economics

.....

.....

Prof. (PhD) Yücel Öztürkoğlu
Director of the Graduate School

ABSTRACT

TASK AND PATH PLANNING IN MULTIROBOT SYSTEMS

Ok, Ahmet Enes

MSc, Artificial Intelligence Engineering

Advisor: Assist.Prof.(PhD) Deniz Özsoyeller

August 2025

Wireless Sensor Networks (WSNs) have been utilized in a wide range of applications such as environmental monitoring, precision agriculture, industrial automation, disaster response, and military surveillance. A critical challenge that affects the system lifetime is the limited battery capacity of the sensors which is mostly consumed by data transmission. In this thesis, we study the sensor data collection problem with a team of unmanned aerial vehicles (UAVs) and a single unmanned ground vehicle (UGV). The UAVs work as data collectors while the UGV provides recharging and data-relaying services to the UAVs. We consider partitioned WSN settings, where the communication graph on the initial locations of all sensors is not connected. That is, a sensor can be single or a member of networked sensors.

Each UAV has a fixed capacity that it reserves for storing sensor data. The objective is to minimize the time required for UAVs to collect the data of all sensors such that no UAV exceeds its sensor data capacity. To solve this problem, we propose a capacity-constrained data collection algorithm that plans energy-aware paths for UAVs so that they cooperatively locate the sensors and collect data from them. The algorithm also determines the UAV-UGV rendezvous locations and plans the UGV's path.

Our approach combines the techniques including environment decomposition, clustering the sensors, and solving the capacitated vehicle routing problem (CVRP). In simulations, we validate the performance of our algorithm varying the key performance metrics including the number of UAVs, number of sensors, communication range, and environment size.

Keywords: Data Collection, Wireless Sensor Networks, Unmanned Aerial Vehicles, Unmanned Ground Vehicle



ÖZ

ÇOK-ROBOTLU SİSTEMLERDE GÖREV VE YOL PLANLAMASI

Ok, Ahmet Enes

Yüksek Lisans, Yapay Zeka Mühendisliği

Danışman: Dr. Öğr. Üyesi. Deniz Özsoyeller

Ağustos 2025

Telsiz Duyarga Ağları (TDA'lar), çevresel izleme, hassas tarım, endüstriyel otomasyon, afet müdahalesi ve askeri gözetim gibi çok çeşitli uygulamalarda kullanılmaktadır. Sistem ömrünü etkileyen kritik bir zorluk, sensörlerin sınırlı pil kapasitesidir ve bu kapasite çoğunlukla veri iletimi sırasında tüketilmektedir. Bu çalışmada, bir grup insansız hava aracı (İHA) ve tek bir insansız kara aracı (İKA) ile sensör verisi toplama problemini ele alıyoruz. İHA'lar veri toplayıcı olarak görev yaparken, İKA İHA'lara şarj ve veri aktarma hizmeti sunar. Çalışmamızda, tüm sensörlerin başlangıç konumlarına göre oluşturulan iletişim grafiğinin bağlantısız olduğu bölünmüş WSN senaryolarını dikkate alıyoruz. Yani, bir sensör tek başına ya da ağ halinde bağlı sensörlerin üyesi olabilir.

Her İHA, sensör verilerini depolamak için ayırdığı sabit bir kapasiteye sahiptir. Amaç, hiçbir İHA bu veri kapasitesini aşmadan, tüm sensörlerin verilerinin toplanması için gereken süreyi en aza indirmektir. Bu problemi çözmek için, İHA'ların sensörleri bulup verilerini toplamak üzere enerji farkındalığına sahip yollar planlamasını sağlayan, kapasite kısıtlı bir veri toplama algoritması öneriyoruz. Algoritma aynı zamanda İHA-İKA buluşma noktalarını belirler ve İKA'nın rotasını planlar.

Yaklaşımımız; alanı bölgelere ayırma, sensörleri kümeleme ve kapasite kısıtlı araç rotalama problemi (KKARP) çözüm tekniklerini birleştirmektedir. Yaptığımız benzetimlerde, algoritmamızın performansını; İHA sayısı, sensör sayısı, iletişim menzili ve ortam boyutu gibi temel performans kriterleri değiştirerek değerlendiriyoruz

Anahtar Kelimeler: Veri Toplama, Telsiz Duyarga Ağları, İnsansız Hava Araçları, İnsansız Kara Aracı



ACKNOWLEDGEMENTS

I would like to begin by expressing my heartfelt appreciation to my advisor, Assist. Prof. Dr. Deniz Özsoyeller, for her constant support, insightful guidance, and patience throughout the course of my graduate studies. Her knowledge and encouragement have played a key role in the completion of this thesis.

I am also truly thankful to my family for their endless understanding, motivation, and emotional support during this journey.

AHMET ENES OK

İzmir, 2025



TEXT OF OATH

I declare and honestly confirm that my study, titled “TASK AND PATH PLANNING IN MULTIROBOT SYSTEMS” and presented as a Master of Science Thesis, has been written without applying any assistance inconsistent with scientific ethics and traditions. I declare, to the best of my knowledge and belief, that all content and ideas drawn directly or indirectly from external sources are indicated in the text and listed in the list of references.

Ahmet Enes OK

10/09/2025



TABLE OF CONTENTS

JURY APPROVAL PAGE	iii
ABSTRACT.....	v
ÖZ.....	ix
ACKNOWLEDGEMENTS	xiii
TEXT OF OATH	xv
TABLE OF CONTENTS.....	xvii
LIST OF FIGURES	xxi
SYMBOLS AND ABBREVIATIONS	xxv
1. CHAPTER: INTRODUCTION	1
2. CHAPTER: RELATED WORK.....	7
2.1. Data Collection in single/multi-UGV systems.....	7
2.2. Data Collection in Single-UAV Systems.....	8
2.3. Data Collection in Multi-UAV Systems	8
2.4. Data Collection in UAV-UGV included Heterogeneous Systems.....	9
2.5. Clustering Approach for Data Collection	9
3. CHAPTER: PROBLEM FORMULATION AND MATHEMATICAL MODEL	11
4. CHAPTER: SOLUTION METHOD	13
4.1. Stage-1: Determining Cluster Heads.....	13
4.2. Stage-2: Exploration.....	16
4.3. Stage-3: Data Collection	17
4.4. Rendezvous Sites For UAV Recharging.....	18
4.5. Example Execution	18
5. CHAPTER: SIMULATIONS AND RESULTS	25
6. CHAPTER: CONCLUSION.....	37

7. REFERENCES	39
APPENDICES	43
7.1. APPENDIX 1. CLUSTERING APPROACH IN STAGE-3 OF THE ALGORITHM.....	43
7.2. APPENDIX 2. CLUSTER HEAD DATA SPLITTING.....	45
7.3. APPENDIX 3. BOUSTROPHEDON PATH GENERATION.....	46
7.4. APPENDIX 4. FOLLOWING THE BOUSTROPHEDON PATH.....	47
7.5. APPENDIX 5. EXAMPLE - OUTPUT OF STAGE-2.....	49
7.6. APPENDIX 6. EXAMPLE - LIST OF THE FOUND SENSORS DURING STAGE-2.....	50
7.7. APPENDIX 7. EXAMPLE – THE SET OF CLUSTERS AFTER STAGE-1	51
7.8. APPENDIX 8. EXAMPLE - CVRP ROUTE GENERATION PHASE ...	52
7.9. APPENDIX 9. EXAMPLE - THE ROUTES AND DISTANCE TRAVELED BY THE UAVS IN STAGE-3	53

LIST OF FIGURES

Figure 1.1. Example System Configuration.....	4
Figure 3.1. Example setup for $n = 2$ UAVs.....	12
Figure 4.1. Example Connected Graphs after phase-1 of Stage-1.	14
Figure 4.2. Pseudocode of CDC: Stage-1	15
Figure 4.3. Stage-1 of the CDC Algorithm.....	19
Figure 4.4. Beginning Phase of Stage 2.	20
Figure 4.5. Middle Phase of Stage 2	21
Figure 4.6. Final Phase of Stage 2.....	22
Figure 4.7. Stage-3: Paths of the UAVs.....	23
Figure 4.8. Stage-3: UGV Path, Recharging Positions and CHs' Data Units.....	24
Figure 5.1. Total Distance Changes over A, k.	27
Figure 5.2. Total Distance Changes over A, n and k.	27
Figure 5.3. Total Distance Changes over R and k.....	28
Figure 5.4. Total Time Changes over A, k.....	28
Figure 5.5. Total Time Changes over A, n and k.	29
Figure 5.6. Total Time Changes over R and k.	29
Figure 5.7. Total Waiting Time Changes over A, k.....	30
Figure 5.8. Total Waiting Time Changes over A, n and k.	30
Figure 5.9. Total Waiting Time Changes over R and k.	31
Figure 5.10. Total Recharging Count Changes over A, k.	31

Figure 5.11. Total Recharging Count Changes over A, n and k.	32
Figure 5.12. Total Recharging Count Changes over R and k.	32
Figure 5.13. Number of Initial CHs.	33
Figure 5.14. Percentage of Single Sensors over CHs.....	33
Figure 5.15. Number of Single Sensors.	34
Figure 5.16. Total Number of Cluster heads.....	34
Figure 5.17. Number of Clusterheads Changes over UAV Capacities.	35
Figure 5.18. Effect of Hop Threshold Values on Number of CHs.	35

SYMBOLS AND ABBREVIATIONS

ABBREVIATIONS:

UAV: Unmanned Aerial Vehicle

UGV: Unmanned Ground Vehicle

WSN(s): Wireless Sensor Network(s)

DCP: Data Collection Problem

GPS: Global Positioning System

CH: Cluster Head

SC: Sub-Cluster (the set of nodes served by a given CH)

DFS: Depth-First Search

PSO: Particle Swarm Optimization

GA: Genetic Algorithm

G-PSO: hybrid Genetic algorithm–Particle Swarm Optimization

TSP: Traveling Salesman Problem

TSPN: Traveling Salesman Problem with Neighborhoods

MTSP: Multiple Traveling Salesman Problem

CVRP: Capacitated Vehicle Routing Problem

QUBO: Quadratic Unconstrained Binary Optimization

DDPG: Deep Deterministic Policy Gradient

AoI: Age of Information

CDC: Constrained Data Collection

ILP: Integer Linear Programming

PARAMETERS:

r : expansion radius

h : altitude

R : communication range between UAV and sensor

n : number of UAVs

A : environment size

S : stripe size

k : number of sensors

c_u : capacity of each UAV

d : total battery capacity of the UAV

1. CHAPTER: INTRODUCTION

Wireless Sensor Networks (WSNs) consist of sensor nodes, that are spatially distributed across a target environment (Yick et al., 2008a). A sensor node is typically equipped with one or more sensors (e.g. for temperature, humidity, pressure, light, vibration, sound, chemical detection), a microcontroller for data processing, limited memory, a wireless transceiver for communication, and a power supply (commonly a battery). Sensor nodes are often designed to be small, low-cost, and capable of self-organizing into ad hoc multi-hop networks. The main goal of a WSN is monitoring and recording the physical conditions of the environment in real time and relay this information to a data sink for analysis and decision-making.

WSNs have been widely adopted for a variety of applications. In environmental and ecological monitoring, they are deployed to track climate parameters, forest fire risks, animal movements, or air and water quality. In precision agriculture, WSNs help in optimizing irrigation, fertilization, and pest control. Industrial automation leverages WSNs for equipment monitoring, fault detection, and predictive maintenance. Disaster response, structural monitoring of buildings and bridges, military surveillance, and habitat monitoring are other notable application domains of WSNs (Kandris et al., 2020).

Despite their usage in versatile applications, WSNs face critical operational challenges (Chong et al., 2003). An important challenge that affects the lifetime of WSNs is energy constraint since sensor nodes are typically battery-powered and usually deployed in locations where battery replacement or recharging is impractical. The limited energy of the sensors is mostly consumed by data transmission. When the sensors complete their data collection task, they form a connected network to forward the gathered sensory data to the data sink. Multi-hop routing is a commonly used method for data relaying. However, an important issue in this method is that the

sensors within close proximity to the sink tend to run out of energy faster than the other sensors. This is because the packets with sink as their destination should be forwarded to these nodes. Furthermore, when the field of interest is large, there can also be single (isolated) sensors. Therefore, in such scenarios, maintaining connectivity among all sensors can only be achieved utilizing many relay (intermediate) nodes, since the sensor nodes can communicate over short distances.

Another approach for data collection other than building a dense static network is to leverage the mobility and flexibility advantages of mobile data collectors (robots) such as Unmanned Aerial Vehicles (UAVs), Unmanned Ground Vehicles (UGVs), or heterogeneous UAV-UGV systems. The robot's ability to move in close proximity to the sensor node to download its sensed data improves the lifetime of the system through reducing (1) the communication burden of the sensors, hence the energy consumption, since deploying many relay nodes would not be needed to guarantee network connectivity, and (2) the data loss rate, thus the number of retransmissions. The approaches based on mobile data collectors often focus on optimizing the UAV's trajectory to minimize energy consumption. Some frameworks employ a single UAV navigating predetermined or dynamically generated paths, while others consider multi-UAV systems that distribute the data collection workload to reduce task completion time or enhance coverage.

In addition to direct data collection strategies, clustering mechanisms are frequently utilized to improve network scalability and energy efficiency (Wang et al., 2018), (Wu et al., 2019). In such schemes, sensor nodes are divided into clusters with designated cluster heads responsible for aggregating and forwarding data to mobile sinks. UAVs or UGVs may then collect data primarily from these cluster heads, with clustering algorithms. Metaheuristic and combinatorial optimization techniques, including genetic algorithms, particle swarm optimization, and capacitated vehicle routing formulations, are commonly applied jointly to solve clustering and path planning problems.

Heterogeneous UAV-UGV frameworks are increasingly investigated to overcome the limitations of employing either aerial or ground-based data collection systems alone. Such systems are especially beneficial in applications such as search and rescue, disaster response, and environmental monitoring (Tokekar et al., 2016), (Singh et al., 2023). In such hybrid architectures, UAVs are often utilized for rapid area coverage, high-resolution aerial mapping, remote sensing, obstacle detection, and data collection from sensors deployed in large environments. Meanwhile, UGVs are deployed to assist when UAV operation is restricted due to line-of-sight, payload limitations, or limited battery endurance. In these systems, UAVs and UGVs may operate independently or cooperatively. Coordination between aerial and ground platforms typically relies on the communication between them, enabling dynamic task allocation and rendezvous planning (Gielis et al., 2022). Path planning strategies in hybrid frameworks range from classical graph-based algorithms (e.g., Dijkstra, A*) and coverage path planning (e.g., boustrophedon and spiral motion patterns), to genetic algorithms, ant colony optimization, or reinforcement learning.

In this thesis, we formulate and study the problem of data collection from k stationary wireless sensors that are distributed over a large area with n capacity-constrained UAVs. Each UAV has a limited communication range and a fixed capacity reserved for storing sensor data. During the mission, a single UGV provides recharging service to the UAVs also assists with relaying information between them. We consider a partitioned WSN scenario where the network among all sensors is not connected. That is, a sensor may be single (isolated) or a member of networked sensors (see Figure 1.1). Moreover, the locations of the sensors are not known by the UAVs and the UGV in advance, and no sensor knows the locations of all sensors. The objective is to minimize the time required to collect the data of k sensors with n UAVs such that each UAV fully fills its sensor data capacity but not exceeds it.

The existing works in the literature for the Data Collection Problem (DCP) mostly focused on the single-UAV systems (Bouhamed et al., 2020; Chen et al., 2020; Gong et al., 2018; Gul et al., 2022; Liu et al., 2024; Luo et al., 2020; Raju et al., 2024; Rezende et al., 2020; Wu et al., 2018) rather than multi-UAV (Amer et al., 2024;

Caillouet et al., 2019; Pan et al., 2023; Wang et al., 2020) or heterogeneous multi-robot systems including UAV-UGV collaboration (Deng et al., 2025; Gielis et al., 2022; Singh et al., 2023).

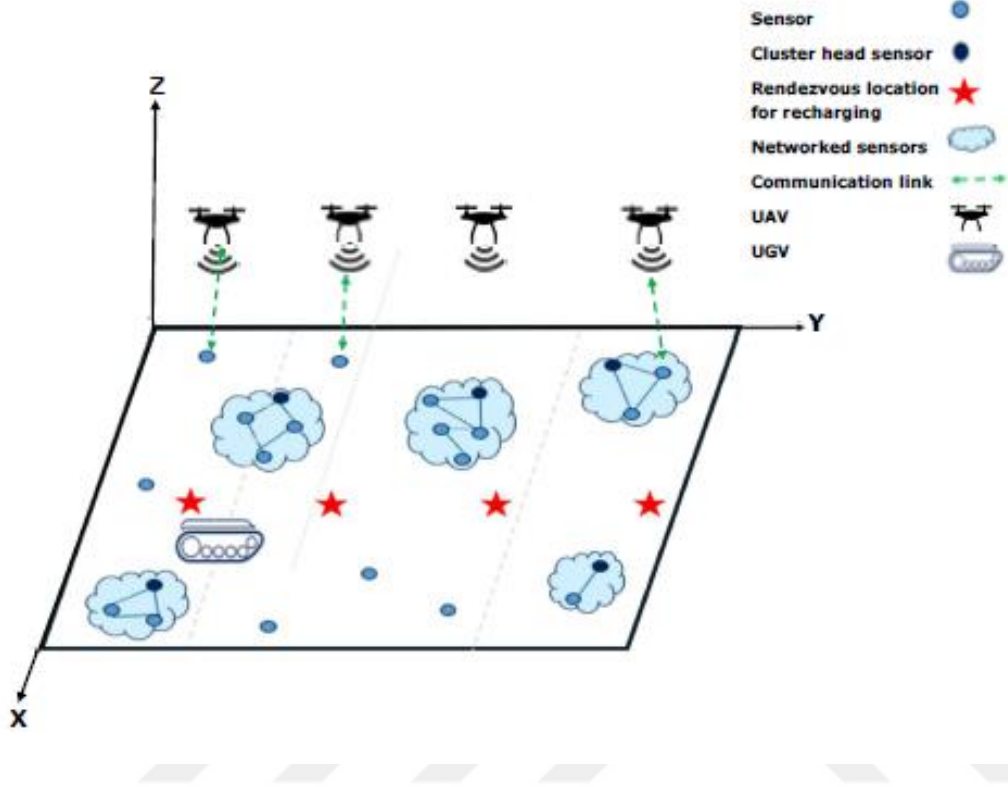


Figure 1.1. Example system configuration with $n = 4$ UAVs, 1 UGV, and $k = 23$ sensors. There are 6 single sensors and 5 networked sensors.

Source: Author

Unlike our work, the studies handling the multi-UAV case consider that the locations of the sensors are known by the UAVs in advance, and do not combine the planning of the UAVs that are subject to both energy and capacity constraints with the planning of a UGV for recharging the UAVs. Moreover, in multi-UAV and UGV case, the UGV participates in the data collection mission rather than providing recharging service to the UAVs or assisting in relaying data between the UAVs as in our work.

Our contributions are as follows. To solve the problem formulated, we propose a Capacity-constrained Data Collection algorithm, named CDC, that plans energy-aware paths for UAVs so that they cooperatively locate the sensors and collect data from them while satisfying their capacity constraint. To minimize the mission time through

efficient cooperation, our algorithm combines the methods including decomposition of the environment among the UAVs, clustering the sensors and determining cluster heads, and solving CVRP. Our solution approach also handles the planning of the rendezvous sites for UAV recharging and UAV-UGV information exchange. In simulations, we validate the performance of our algorithm by varying the key evaluation metrics including the number of UAVs, number of sensors, communication range, and environment size.

The rest of the article is organized as follows. We begin by presenting the related works in Section 2. The problem studied is formulated in Section 3. Our proposed algorithm and its example execution are presented in Section 4. We report the results of the simulations conducted in Section 5. Finally, the conclusion including future work is presented in Section 6.

2. CHAPTER: RELATED WORK

Data Collection in wireless sensor networks (WSNs) is a well-explored yet continuously evolving field, where challenges such as energy consumption, latency, security, and adaptability to real-world settings still exist. The integration of aerial and ground robots has led to the design of diverse architectural and algorithmic approaches. Here, we present the related work by organizing it into five categories: Data Collection in single/multi-UGV systems (Section 2.1), single-UAV Systems (Section 2.2), multi-UAV Systems (Section 2.3), UAV-UGV included heterogeneous systems (Section 2.4), and clustering approach for data collection (Section 2.5).

2.1. Data Collection in single/multi-UGV systems

Utilizing ground robots (data mules) for data collection in WSNs has been extensively studied (Bhadauria et al., 2011; Chen et al., 2011; Chiu & Liu, 2011; Hu et al., 2022; Liu et al., 2013; Najjar-Ghabel et al., 2020; Papatheodorou et al., 2018; Tsilomitrou et al., 2018; Tsilomitrou & Tzes, 2021). Tsilomitrou et al. (2018) considered mobile robots as data mules, optimizing minimal delay and route cost, while Liu et al. (2013) and Papatheodorou et al. (2018) explored clustering-based and game-theoretic algorithms for path planning and data retrieval by mobile sinks. Najjar-Ghabel et al. (2020) addressed DCP in an obstacle-populated environment proposing an algorithm that clusters the nodes and utilizes a mobile sink for data gathering from the cluster heads. Chen et al. (2011) studied the DCP in the partitioned/islanded WSNs with a mobile robot. The robot collects data from isolated segments based on energy thresholds and plans paths using heuristic methods like dynamic moving-based scheduling. Tsilomitrou and Tzes (2021) studied the DCP with multiple robots and stationary sensors, formulating the problem as a variant of traveling salesman subset-tour problem. Various constraints (e.g. pairwise distance between the sensors, visiting time, data download time) were investigated. Bhadauria et al. (2011) studied DCP with multiple ground robots both when the robots' motion is restricted to a curve also in 2D

environments. The authors presented an optimal algorithm for the first case and a constant factor approximation algorithm for the latter case.

2.2. Data Collection in Single-UAV Systems

Single-UAV systems for data collection have been the focus of numerous studies (Bouhamed et al., 2020; Chen et al., 2020; Gong et al., 2018; Gul et al., 2022; Liu et al., 2024; Luo et al., 2020; Raju et al., 2024; Wu et al., 2018). Bouhamed et al. (2020) studied autonomous data gathering in delay-tolerant WSNs using a single multi-rotor UAV in a 3D obstacle-populated area. The UAV, starting and returning to a charging station as needed, collects data directly from static ground sensor nodes. Their system integrates deep reinforcement learning techniques: DDPG for continuous 3D obstacle-avoiding navigation and Q-learning for scheduling sensor visits considering node priority, battery, and charging constraints. Gong et al. (2018) addressed data collection with a single UAV tasked with gathering information from linearly deployed sensors (e.g. along roads). The authors focused on minimizing total flight time through joint optimization of UAV speed, trajectory segments, and sensor transmit power, and proposed solutions based on dynamic programming and water-filling/bisection search for multi-sensor and single-sensor cases, respectively. Luo et al. (2020) studied single-UAV data collection in WSNs where the sensors in WSN are randomly deployed in a 2D plane. The UAV hovers within each sensor's data region to collect data. They formulated latency and data collection ratio optimization as NP-Complete problems, proposing a solution based on traveling salesman with Neighborhoods problem (TSPN) and greedy approximation algorithms for path planning. Raju et al. (2024) focused on an urban flood monitoring scenario using a single UAV that partially collects data during each pass to balance energy and Age of Information (AoI). Their solution jointly optimizes data collection volume and hovering times via a hybrid genetic algorithm–particle swarm optimization (G-PSO) method.

2.3. Data Collection in Multi-UAV Systems

Instead of a single UAV, multiple UAVs can carry out the data collection task (Amer et al., 2024; Caillouet et al., 2019; Pan & Fu, 2023; Wang et al., 2020). Caillouet et al. (2019) explored continuous coverage of mobile ground sensors with a fleet of UAVs deployed in a 3D area. Data is relayed through multihop air-to-air communication to a

base station. Their work minimizes UAV count, altitudes, and movement cost over time using mixed integer linear programming and column generation. Pan and Fu (2023) proposed a collaborative multi-UAV data collection model in partitioned IoT regions. UAVs assigned to remote zones relay data to proximity UAVs, who return it to the base station. The system uses genetic algorithms for path planning and workload balancing, statically selecting a closest node in each sensor group as a data aggregation point, focusing on minimizing AoI. Wang et al. (2020) addressed DCP in WSNs that are deployed in a 2D environment utilizing multiple UAVs. The UAVs dynamically switch between flying and hovering modes depending on sensor energy levels and data requirements. The system formulates UAV-sensor association and path planning as a multiple traveling salesman problem (MTSP) and solved it applying a genetic algorithm. Amer et al. (2024) introduced a multi-UAV energy-optimized data collection framework for large-scale WSNs using triangulation-based K-means clustering for aggregator placement, the gaining-sharing knowledge (GSK) optimization algorithm for dockstation/aggregator power optimization, and CVRP-based trajectory planning.

2.4. Data Collection in UAV-UGV included Heterogeneous Systems

To minimize energy consumption and maximize data collection using a UAV and a UGV, Singh et al. (2023) proposed a modified Greedy Randomised Adaptive Search algorithm. Deng et al. (2025) studied DCP with multiple UGVs and UAVs over large areas and proposed a scheme that allocates regions to each UGV and assigns a number of UAVs to each region. A multi-agent deep deterministic policy gradient algorithm was proposed to optimize the routes of the UGVs and UAVs.

2.5. Clustering Approach for Data Collection

Clustering approach is used in many works for improving scalability and energy efficiency. Abu-Baker et al. (2023) considered a single rotary-wing UAV system for data collection in 3D environments, where wireless sensor nodes are organized into clusters. Cluster heads (CHs) are selected based on residual energy, proximity to cluster centroid, and minimal path loss, with clustering performed via particle swarm optimization (PSO). The UAV's 3D placement is optimized using genetic algorithms (GA) to maximize network lifetime and improve uplink data transmission efficiency.

Dios et al. (2017) presented an integrated system where fixed-wing UAVs autonomously deploy, localize, and collect data from sensor nodes over large-scale areas. Sensors self-organize into clusters with CHs selected based on energy and link quality. Wang et al. (2018) introduced UAV-assisted CH election based on node connectivity for robust aggregation, while Wu et al. (2019) and Batta et al. (2021) reviewed and simulated multi-hop clustering strategies. Tang et al. (2024) focused on strip-shaped networks with branches, proposing clustering and path planning mechanisms tailored to linear topologies to balance energy consumption and delay. Liu et al. (2024) proposed an energy-efficient routing protocol for UAV-WSNs where a single UAV collects data via multi-hop communication with CHs. Their system uses unequal-sized dynamic clusters, CH selection based on residual energy and proximity, and Grey Wolf Optimizer for UAV collection position adjustment.

3. CHAPTER: PROBLEM FORMULATION AND MATHEMATICAL MODEL

Consider n identical UAVs and a single UGV that are operating in a square-shaped bounded environment A with a side length of A_w . The UGV has a recharging station placed on it, thus can act as a mobile recharger. The UAV is capable of landing on this station to be recharged and then taking off and proceeding with its task.

The UAVs and UGV are initially located at a common depot D_0 . There are k identical stationary sensors that are located within A . We do not assume that there is a connected network among all sensors. Therefore, a sensor can be a single or non-single, that is, a member of a networked sensors called *cluster*. The UAVs, UGV, and sensors are equipped with a Global Positioning System (GPS). The UAVs and UGV do not know the locations of the sensors in advance. Moreover, a single sensor does not know the location of any other sensor, whereas a non-single sensor knows only the locations of the sensors within its cluster, but not the ones outside it.

Let $\mathcal{S} = \{s_1, \dots, s_k\}$ denote the set of sensors and $\mathcal{U} = \{u_1, \dots, u_n\}$ denote the set of UAVs. The set of the locations of the sensors is denoted by $\mathcal{P}_s = \{p_1^s, \dots, p_k^s\}$. Each sensor initially has c_s units of data to be collected, and each UAV has a maximum storage capacity of c_u units that it reserves for sensor data. We consider that $c_u \geq c_s$, $k \geq n$, and $kc_s = nc_u$. UAV u_i should visit sensor s_j and hover above it to collect the gathered data of s_j , where $i \in \{1, \dots, n\}$ and $j \in \{1, \dots, k\}$. Note that the UGV aids the UAVs' mission by recharging them and relaying information between them about the found sensors. However, it neither involves in nor reserves a storage capacity for collecting sensor data.

Any pair of sensors can communicate with each other and is said to be connected if the distance between them is within R . The UAV can communicate with a sensor only if the distance between them is within $R = \sqrt{r^2 + h^2}$, where r is the expansion radius (communication footprint) and h is the altitude of the UAV. The UAVs fly at a fixed altitude h . Let $S = A_w/n$ and $S_v = S/2r$, where $n|A_w$ and $2r|S$ (see Figure 3.1). Both

the UAVs and UGV move at unit speed v . Thus, we use time and distance interchangeably. The UAVs fly at a fixed altitude h and initially have full battery capacity.

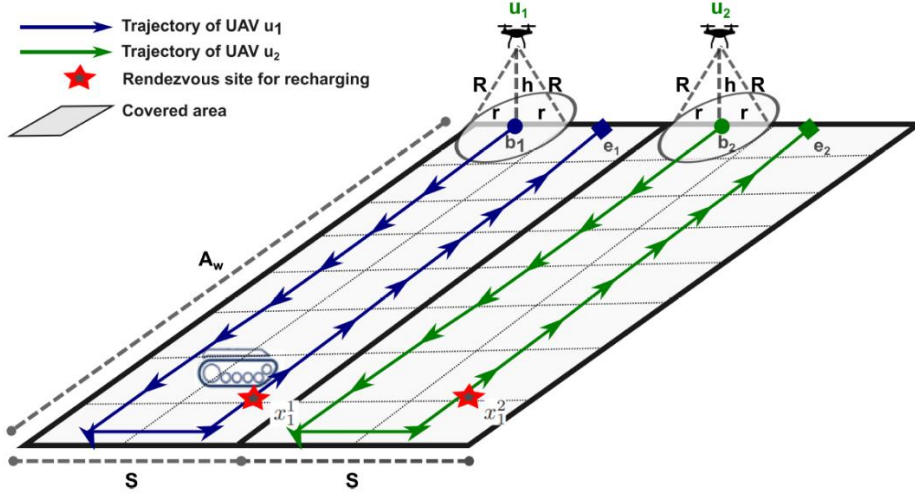


Figure 3.1. Example setup for $n = 2$ UAVs. x_1^1 and x_1^2 are the rendezvous sites.

Source: Author

We make the following assumptions.

1. The UAV's battery discharges at a unit rate (one unit per unit distance traveled).
2. The maximum distance units UAV can spend with 100% battery capacity is equivalent to d after which it requires recharging by the UGV.
3. The required time to collect c_s units of sensor data is fixed and denoted by T .
4. The UGV has unlimited battery capacity.

Problem: The objective is to minimize the time required to collect the data of all sensors in $\mathcal{S}$ with n UAVs such that c_u for each UAV in $\mathcal{U}$ includes $(kc_s)/n$ units of sensor data; we refer this constraint on capacity as C1.

4. CHAPTER: SOLUTION METHOD

In this section, we present our Capacity-constrained Data Collection algorithm CDC which includes 3 main stages: Stage-1: Determining cluster heads, Stage-2: Exploration, and Stage-3: Data collection. The goal of Stage-1 is to determine the cluster head sensor nodes for the clusters (i.e. WSNs) in A . In Stage-2 and Stage-3, the UAVs collaboratively explore A and collect the data of all sensors, respectively.

4.1. Stage-1: Determining Cluster Heads

Our goal in Stage-1 is decrease the total distance traveled by the UAVs, thus the mission time, through reducing the number of sensors that the UAVs should visit in Stage-2. To achieve this, we use a centralized approach that is composed of two phases: phase-1 and phase-2. The pseudocode of CDC: Stage-1 is given in Algorithm 1.

Given $\mathcal{P}_s$, in phase-1, we determine the clusters in A . Let $G(V, E)$ be an undirected graph, where V represents the vertex set and E represents the edge set. V includes the sensors in $\mathcal{S}$ and E is constructed by adding an edge between any pair of sensor nodes that are within R distance to each other. If $E \neq \emptyset$, then the cluster set $\mathcal{C}$ is obtained by applying a recursive Depth-First Search (DFS) to G . However, if $E = \emptyset$, then there are no clusters in A so phase-2 of Stage-1 is not executed.

In phase-2, we determine the cluster heads (CHs) in each cluster considering c_u and the hop threshold to a sensor in the cluster. Note that a cluster can have more than one CH. The CHs for cluster $C_i \in \mathcal{C}$ are determined as follows. First, the sensors in C_i are sorted, with respect to their number of 1-hop neighbors, into C'_i . Top element in C'_i is selected as the CH and added to the sub-cluster SC . Next, starting with the closest 1-hop neighbors of the CH(SC), the neighbor sensors of CH(SC) are added to SC until the total c_s of the sensor nodes in SC does not exceed c_u and $\kappa \leq \tau$, where $\kappa = 1$ and $\tau \geq 1$ is the hop threshold which can be determined according to the system

requirements. Finally, the sensors nodes in SC are deleted from C'_i . This process continues until C'_i is empty. Since each sensor in SC will forward its data to $CH(SC)$, the number of sensors that the UAVs should visit in Stage-2 decreases.

At the end of Stage-1, a single sensor node still has c_s units of data to be collected, whereas a CH sensor node has $c_s|SC|$ units of data to be collected. We consider that in advance of Stage-2, each sensor knows if it is a CH or not, and each CH has the forwarded data from the sensors in its sub-cluster SC .

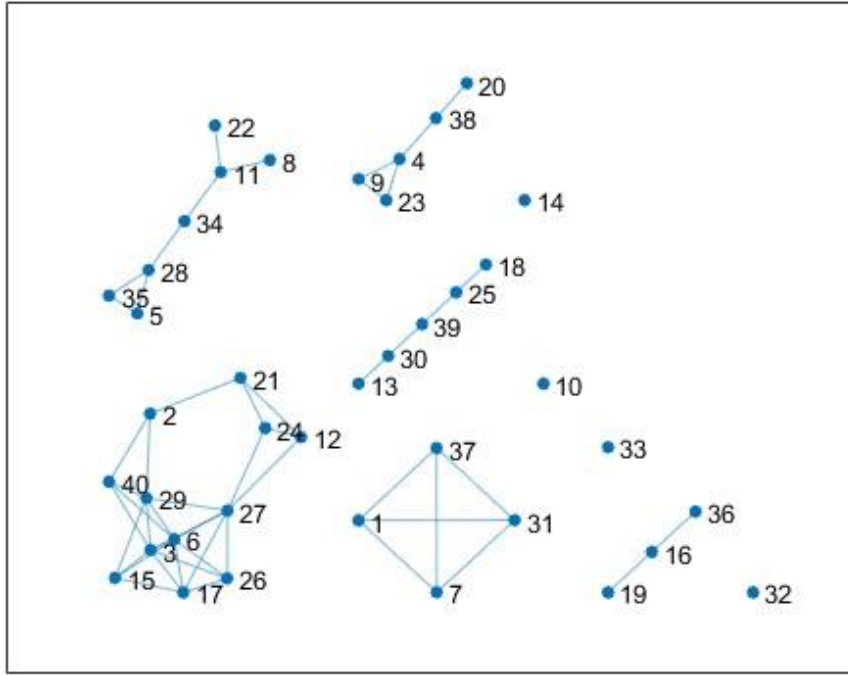


Figure 4.1. Example Connected Graphs after phase-1 of Stage-1.

Source: Author

This figure illustrates the resulting connectivity graph for 40 sensors randomly deployed over a 240×240 area. Edges represent connections between sensors within a communication range of $R = 31.62$ units. The algorithm forms 6 distinct clusters, while 4 sensors (s_{10} , s_{14} , s_{32} , s_{33}) remain isolated, as single nodes, due to communication range constraint. The visualization demonstrates how sensor distribution and network parameters directly affect cluster formation, connectivity, and the presence of isolated nodes. The algorithm (refer to Algorithm 1)- returns the set and number of sensors to

be visited in Stage-3 which are denoted by $\bar{S}$ and $\bar{k}$, respectively. Since each sensor in SC will forward its data to $CH(SC)$, we will have $\bar{k} \leq k$.

Algorithm 1: CDC: Stage-1

Input : $\mathcal{P}_s$: Locations of the sensors in S , τ : hop threshold
Output : $\bar{S}, \bar{k}$

```

// Phase-1: Construct the connectivity graph
on  $\mathcal{P}_s$ 
1 Create undirected graph  $G(V \leftarrow S, E \leftarrow \emptyset)$ 
2 foreach sensor pair  $s_i, s_j \in S$  do
3   if  $dist(p_i^s, p_j^s) \leq R$  then
4      $V = V \cup edge(s_i, s_j)$ 
5   end
6 end
// Phase-1: Construct the clusters
7  $C \leftarrow \emptyset$ 
8 foreach sensor  $s_i \in S$  do
9    $C_i \leftarrow DFS(G, s_i)$ 
10   $C \leftarrow C \cup C_i$ 
11 end
// Phase-2: Determine the CHs and the
// assigned sensors to each CH
12  $count = 0$ 
13  $full = false$ 
14  $\kappa = 1$ 
15  $SC \leftarrow \emptyset$ 
16  $SC_{temp} \leftarrow \emptyset$ 
17  $CHs \leftarrow \emptyset$ 
18 foreach cluster  $C$  do
19    $C_i \leftarrow$  current cluster in  $C$ 
20    $C'_i \leftarrow$  sorted  $C_i$  w.r.t. the number of 1-hop neighbors
21   while  $C'_i \neq []$  do
22      $CH \leftarrow$  top element in  $C'_i$ 
23     while  $count < c_u$  and  $\kappa \leq \tau$  do
24        $N \leftarrow$  closest list of neighbors of  $CH$  at  $\kappa$ -hop
25        $N \leftarrow N \setminus SC$ 
26       foreach sensor  $s_i \in N$  do
27         if  $count < c_u$  then
28            $count = count + c_s$ 
29            $SC_{temp} \leftarrow SC_{temp} \cup s_i$ 
30         end
31         if  $count == c_u$  then
32            $full = true$ 
33           break
34         end
35       end
36       if  $full == false$  then
37          $\kappa = \kappa + 1$ 
38       else
39         break
40       end
41     end
42      $SC \leftarrow SC \cup SC_{temp} \cup CH$ 
43      $\bar{S} \leftarrow \bar{S} \cup CH$ 
44     Delete the sensors in  $SC_{temp}$  from  $C'_i$ 
45   end
46    $N \leftarrow []$ 
47    $SC_{temp} \leftarrow []$ 
48 end
49  $\bar{k} = |\bar{S}|$ 

```

Figure 4.2. Pseudocode of CDC: Stage-1

Source: Author

4.2. Stage-2: Exploration

In this stage, UAVs collaboratively explore A to locate the sensors and identify the cluster head and single sensor nodes. Our approach exploits an exact decomposition of A among the UAVs, which is the union of non-overlapping, equal-sized regions (rectangular stripes) composing A . Each UAV is assigned a stripe to cover. Let b_i and e_i denote the locations where UAV u_i starts and ends covering its stripe, respectively. Before Stage-2 is executed, each UAV is deployed at its starting location and the UGV is deployed at x_1^1 , e.g., this can be done utilizing a deployment vehicle. UAV u_i covers its stripe following a Boustrophedon path π_i , that is, performing back-and-forth motions (Choset & Pignon, 1998). An example of this motion pattern is shown in Figure 3.1.

Let $l(\cdot)$ denote the length of a path. Since the main goal is to locate the sensors and communicate with them as they are found, the expansion radius r between the vertical paths in π_i , thus the length of π_i , is determined according to R . Whenever u_i finds a sensor, it keeps the sensor's location and state (CH/single) through communication.

The rendezvous site where u_i is recharged by the UGV for the j -th time is denoted by x_j^i . When u_i requires recharging for the j -th time, it waits for the UGV at x_j^i . Each time the UGV visits a UAV, it collects the set of sensors that the UAV has found until that time and adds this information to its set F_g . F_i^j denotes the set of found sensors by u_i at the time it arrives at x_j^i . When the UGV meets u_i at x_j^i , u_i sends F_i^j to the UGV which updates its F_g , and in turn receives F_g and updates F_i^j . Once the recharging is completed, u_i continues covering its stripe. The coverage task ends when u_i arrives at e_i or its $F_i^j = S$. If F_i^j becomes equal to S during the traversal of u_i , then u_i stops the exploration at the next rendezvous site.

When the UGV ensures that the UAVs will not further require its assistance for recharging in Stage-2, it visits each UAV at the location where it completed the coverage task. Let f_i denote the location where UAV u_i has terminated Stage-2. When the UGV arrives at f_i , it fully recharges u_i before it starts Stage-3.

4.3. Stage-3: Data Collection

Once the UAVs learn the location, state (CH/single), and sensory data size of all sensors in Stage-2, in Stage-3, they plan their paths to collect data from the sensors. The objective in this stage is to minimize the total length of paths and balance the workload assigned to each UAV while satisfying C1. Therefore, a CVRP formulation is devised for efficient cooperation of the UAVs.

Dantzig et al. (1954) demonstrated that the Vehicle Routing Problem (VRP) is an NP-hard combinatorial optimization problem that generalizes another NP-hard problem well-known as Traveling Salesman Problem (TSP). For a given set of sites, TSP attempts to find the shortest possible route for a single vehicle to visit each site exactly once and return to the origin city, whereas VRP focuses on finding the optimal set of routes for a fleet of vehicles to visit all sites. CVRP is one of the most well-studied variants of VRP (Eksioglu et al., 2009). Its main difference from VRP is that each vehicle now has a limited capacity, so the sum of each site's demand along the route must not exceed the vehicle's capacity.

The approach to solve the CVRP, which we call CVRPsolver, involves creating clusters that represent the set of sites visited by a vehicle in a single route and solving TSP for each cluster. We use the clustering approach proposed in (Feld et al., 2019). Instead of formulating the TSP as a Quadratic Unconstrained Binary Optimization (QUBO) problem as proposed by Feld et al. (2019), the ILP-based TSP solver was used due to its better efficiency in handling large number of sensors (large TSP instances).

Each vehicle in CVRP starts and ends at the same depot D_1 . Therefore, we execute the CVRPsolver defining D_1 as the center of the smallest circle that encloses $f_i \forall u_i \in \mathcal{U}$. Once the path π_i^* of u_i is obtained, we modify the path from D_1 to $\text{first}(\pi_i^*)$ as f_i to $\text{first}(\pi_i^*)$, and the path from $\text{last}(\pi_i^*)$ to D_1 as $\text{last}(\pi_i^*)$ to f_i . When u_i visits sensor s on path π_i^* , it establishes communication with s and collects its data.

We consider that the CVRPsolver is executed by the UGV after its F_g includes the locations of all found sensors. When the UGV visits u_i at f_i at the end of Stage-2, it relays F_g , the obtained π_i^* from the CVRPsolver, and the rendezvous sites on π_i^* to u_i .

4.4. Rendezvous Sites For UAV Recharging

Recall that x_j^i is the rendezvous site that u_i is recharged by the UGV for the j -th time. The set of rendezvous sites for UAV u_i to be recharged by the UGV during the exploration or data collection stage of CDC algorithm is denoted by $X_i = \{x_1^i, \dots, x_{|X_i|}^i\}$ where

$$|X_i| = \begin{cases} \lceil ((A_w + 2r)(S_v - 1))/d \rceil, & \text{for Stg.2,} \\ \left\lceil \left(\sum_{\text{sensor } s \in \pi_i^*} \text{size}(s)T + \ell(\pi_i^*) \right) / d \right\rceil, & \text{for Stg.3.} \end{cases}$$

The size of a sensor is given by

$$\text{size}(s) = \begin{cases} c_s, & \text{if sensor } s \text{ is single,} \\ |SC(s)|c_s, & \text{if sensor } s \text{ is CH.} \end{cases}$$

During the execution of stages 2 and 3 of the CDC algorithm, UAV u_i is only allowed to be recharged at a predetermined rendezvous site in X_i . For the UGV, $|X_i|$ rendezvous sites are placed along the path of u_i at equal distances.

UGV path planning: The UGV divides its service process in n rounds and provides service to each UAV in each round. When the UGV provides service to all UAVs in a round, it proceeds with the next round. In Stage-2, the UGV first visits $x_1^i, \forall u_i \in \mathcal{U}$, then $x_2^i, \forall u_i \in \mathcal{U}$, and so on. In Stage-3, the path of the UGV is planned by solving the TSP using the nearest neighbour algorithm (Jünger et al., 1995).

4.5. Example Execution

An example execution of the CDC Algorithm is presented in Figures 4.3 to 4.8. $k = 60$ sensors are distributed uniformly at random in A of size 240×240 units. $n = 4$ UAVs and 1 UGV are deployed at the boundary of A to carry out the mission. The parameters in this example execution are defined as follows: $h = 10, r = 15, R = 18.02, c_u = 15, c_s = 1, T = 6, d = 340$ and $\tau = 1$.

Figure 4.3 shows the result of Stage-1 execution. There are 24 single sensors and 13 networked sensors (clusters). Clusters $C_1 - C_{13}$ have 1, 1, 1, 2, 1, 1, 3, 1, 1, 1, 1, 1, 1 CHs, respectively. Therefore, in Stage-3, the UAVs collect data from $\bar{k} = 40$ sensors instead of $k = 60$ sensors.

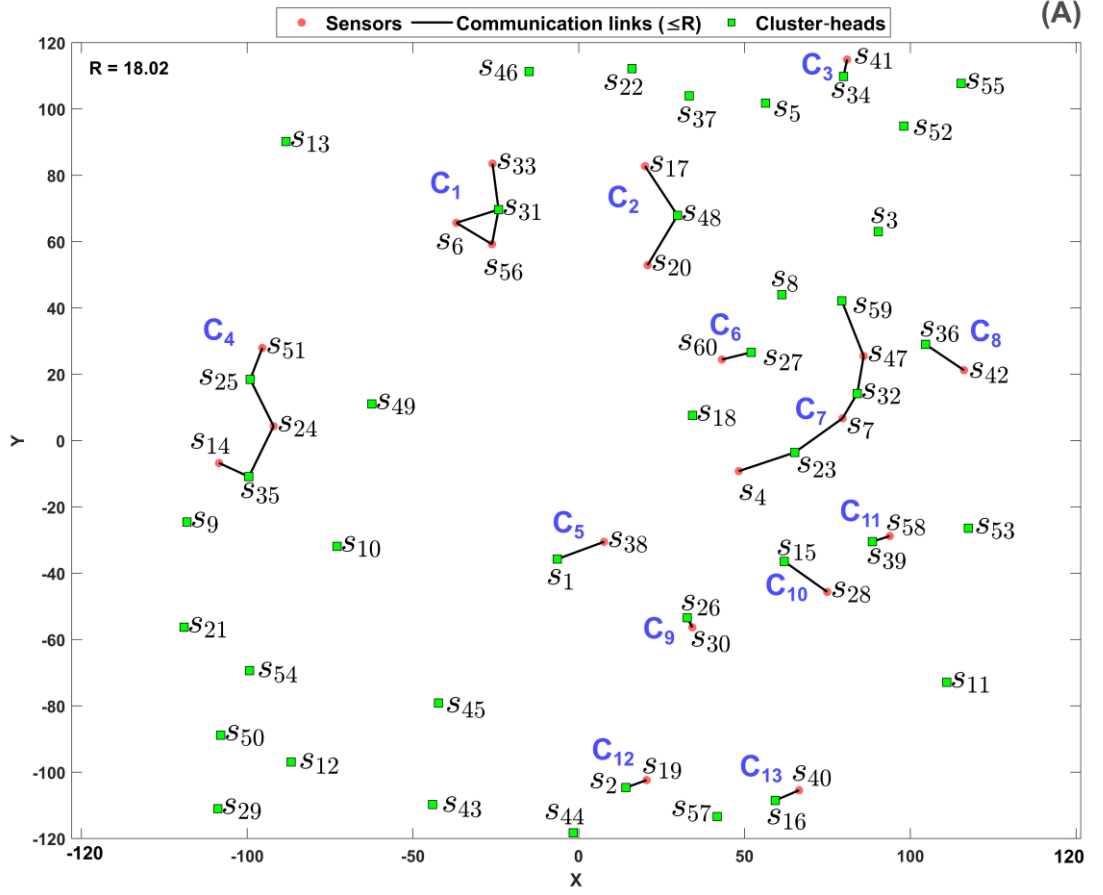


Figure 4.3. Stage-1 of the CDC Algorithm

Source: Author

Figures 4.4 to 4.6 show the sensors that are found and not found and the trajectories of the UAVs at the beginning, middle, and final of Stage-2, respectively. The sensors that are found and not found are shown as green and red circles, respectively. Since $d = 340$, there is only one rendezvous site in each stripe. Figure 4.6 also shows the traversal of the UGV (see the blue dashed line) between the rendezvous sites, and its visit to the UAVs to fully charge them before Stage-3 and inform them about their paths and the rendezvous sites on their paths during Stage-3.

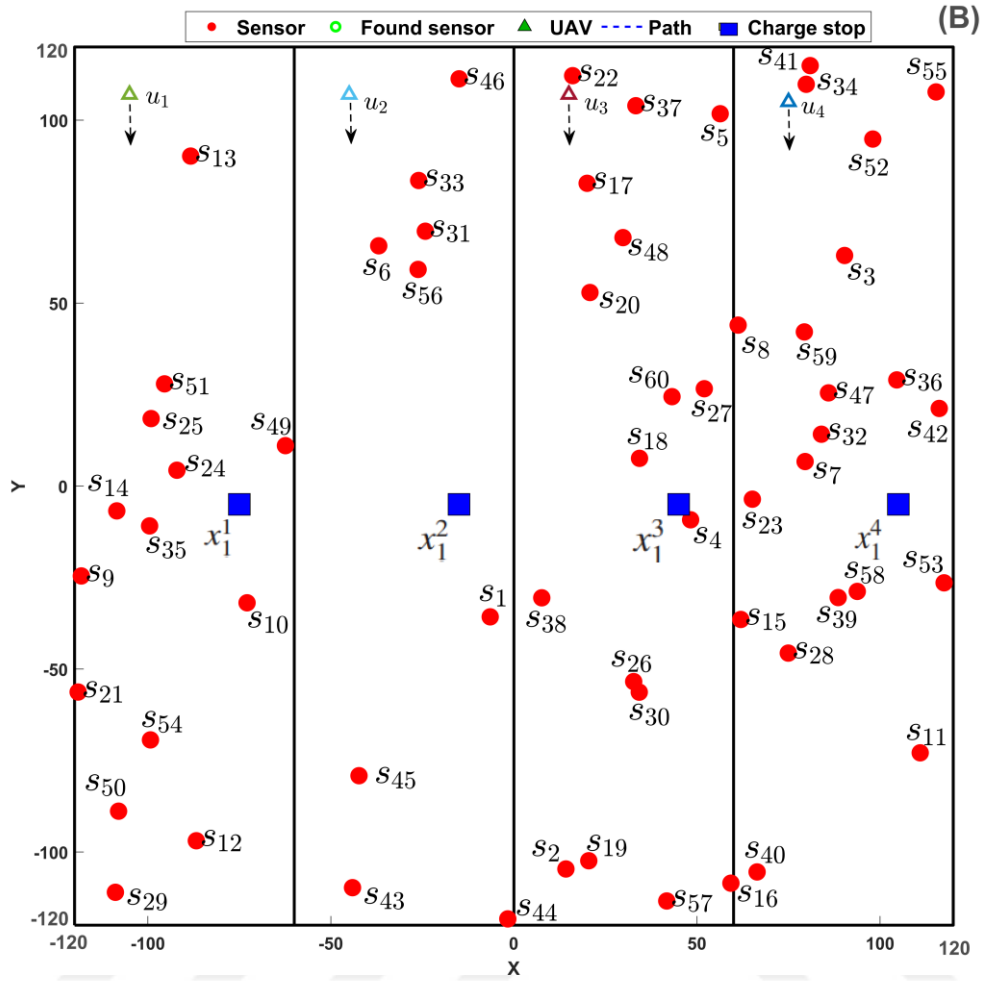


Figure 4.4. Beginning Phase of Stage 2

Source: Author

Figure 4.4 illustrates the initial configuration at the start of Stage-2. It shows the UAV trajectories as they begin their search, along with the status of sensors at this early point. Green circles represent the sensors that have been located by the UAVs, while red circles indicate the ones that have not yet been found. The figure reflects the UAVs' early movements in the environment and the spatial distribution of sensors before complete coverage begins.

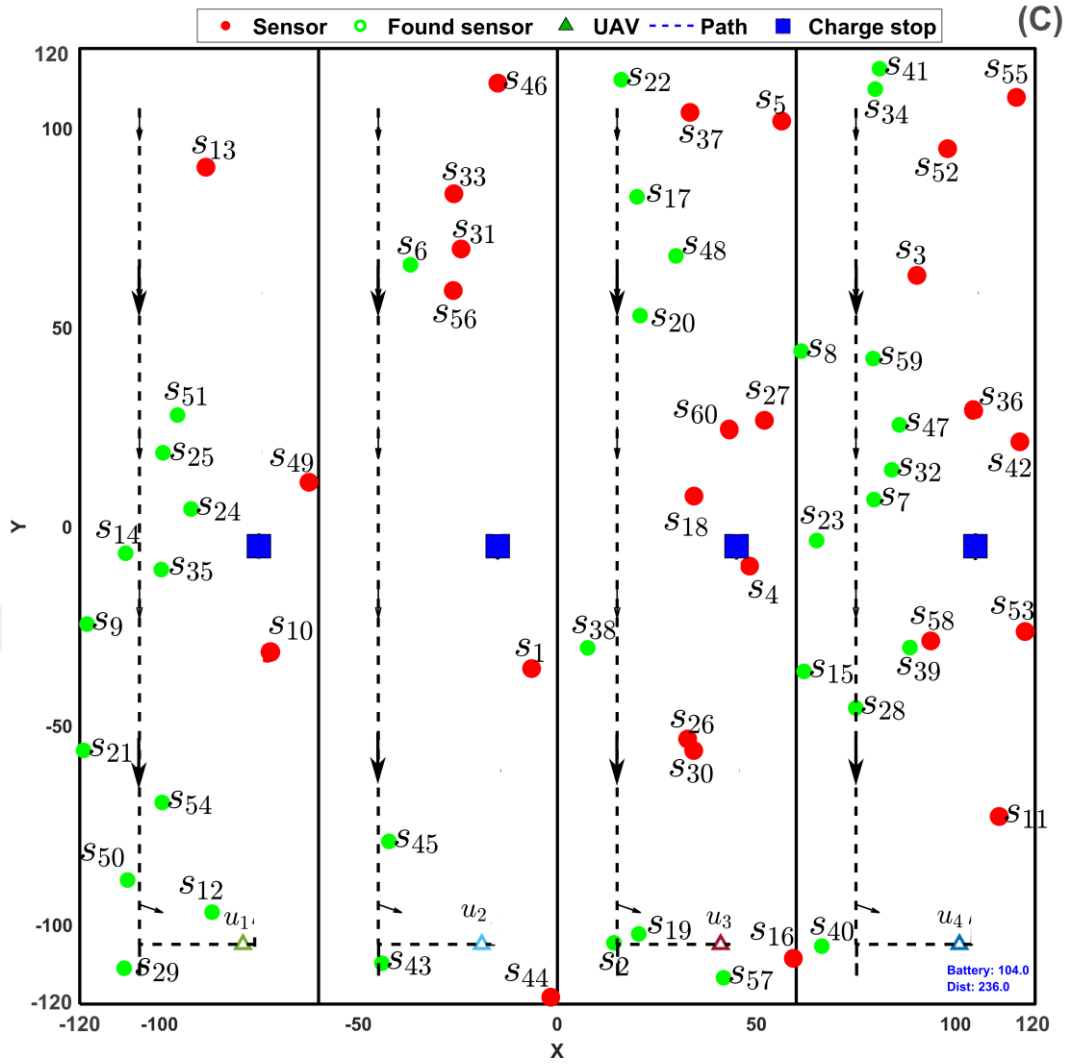


Figure 4.5. Middle Phase of Stage 2

Source: Author

Figure 4.5 shows the UAVs' progress at the midpoint of Stage-2. As the search continues, more sensors have been discovered, increasing the number of green circles compared to the previous stage. The figure highlights the evolving trajectories of the UAVs as they adapt to new discoveries and approach more unexplored areas. Red circles still indicate the sensors that remain undetected at this stage.

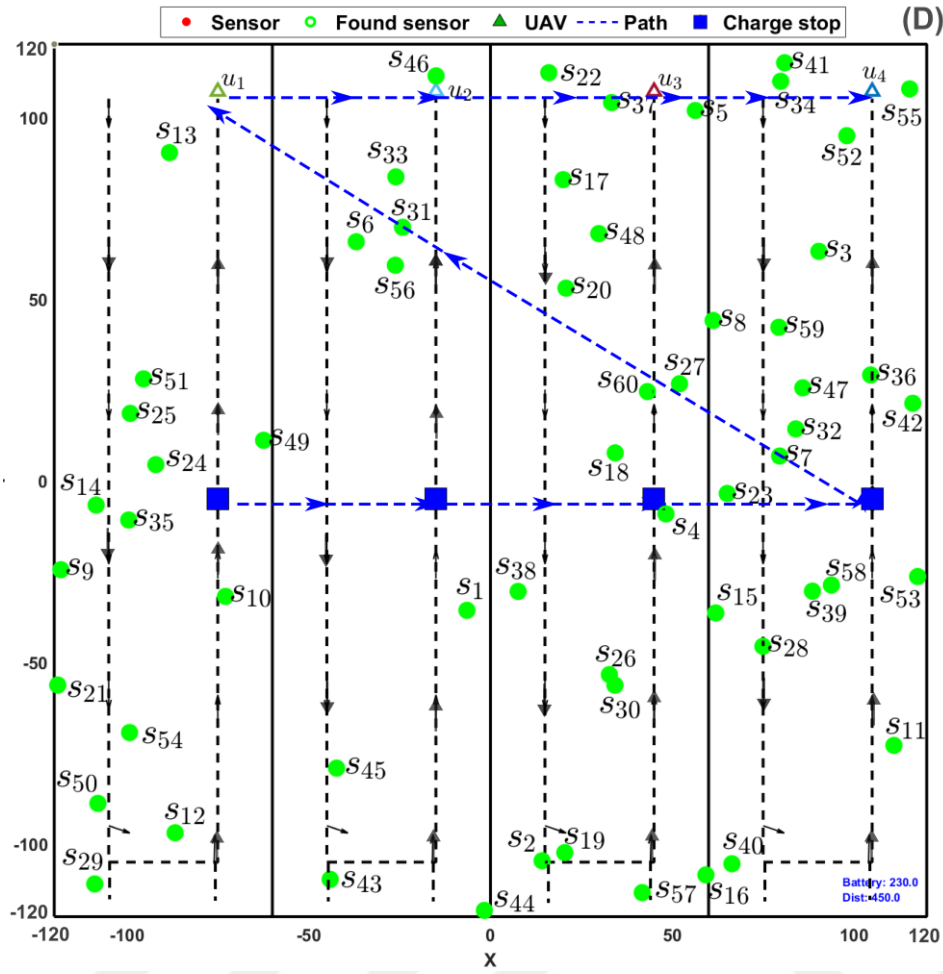


Figure 4.6. Final Phase of Stage 2

Source: Author

Figure 4.6 presents the final state of Stage-2, showing all found and unfound sensors, the completed UAV trajectories, and the movement of the UGV. Green and red circles again mark found and unfound sensors, respectively. Additionally, this figure includes the UGV's traversal path (depicted with a blue dashed line) as it visits each rendezvous site. With $d = 340$, only one rendezvous site exists per stripe. The UGV stops to fully recharge the UAVs and to communicate their Stage-3 paths and assigned rendezvous locations.

Figures 4.7 and 4.8 show the paths of the UAVs in Stage-3. The paths of the UAVs determined by the UGV, with small black circles as the starting points, are presented in Figure 4.8. The number next to each sensor in Figure 4.8 represents the data units that each sensor has and is collected by one of the UAVs. The total data

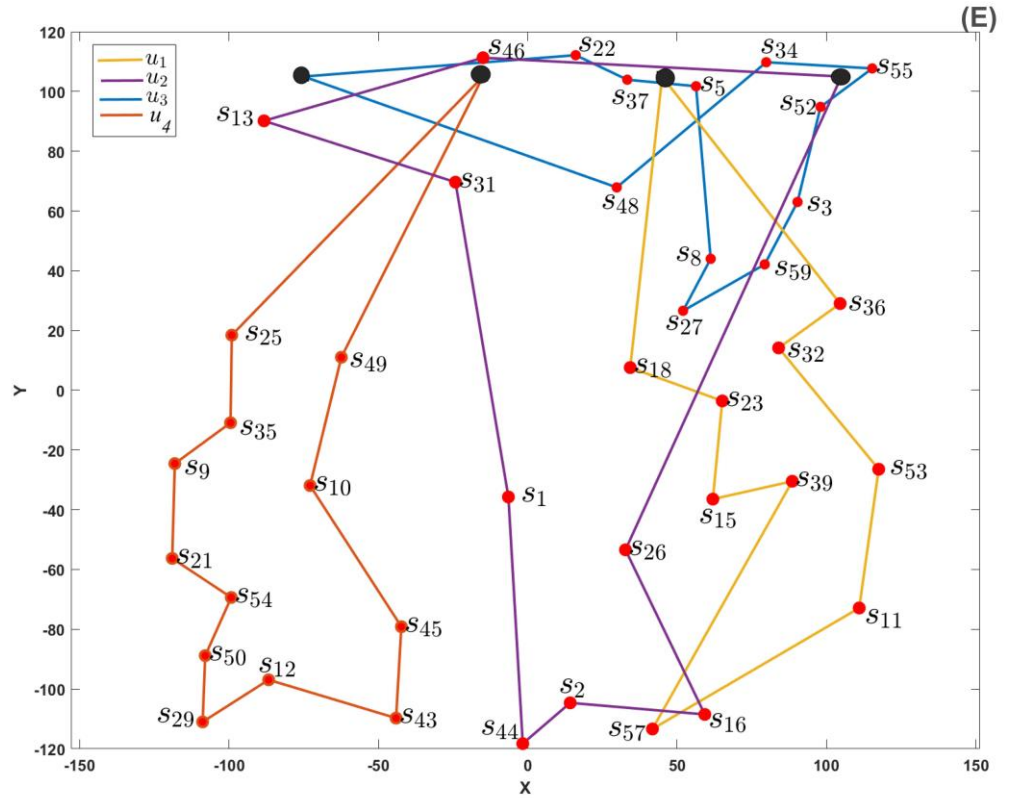


Figure 4.7. Stage-3: Paths of the UAVs

Source: Author

Figure 4.7 displays the initial path plans of all UAVs during Stage-3. Each trajectory outlines how the UAVs are assigned to navigate the environment for data collection based on the UGV's prior coordination. It provides a general overview of the flight paths without specific sensor load details.

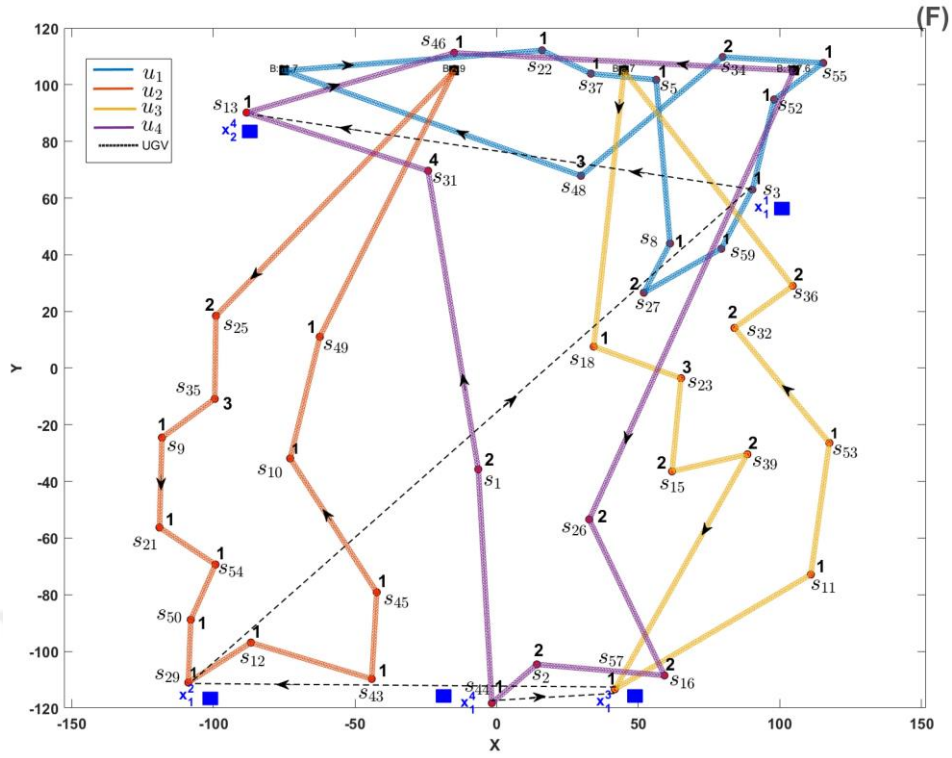


Figure 4.8. Stage-3: UGV Path, Recharging Positions and CHs' Data Units

Source: Author

This figure not only illustrates the assigned trajectories and data loads of each Cluster heads, but also presents the detailed path of the UGV during Stage-3. The UGV's movement between rendezvous sites is explicitly shown, along with the sequence in which it visits and recharges the UAVs. Specifically, in the first service round, the UGV recharges u_4 , u_3 , u_2 and u_1 , in that order. In the second round, only u_4 is visited again. The figure visualizes both the UAV coordination and the UGV's supporting role in enabling continuous data collection.

5. CHAPTER: SIMULATIONS AND RESULTS

In simulations, we evaluate the performance of our algorithm, implementing it in MATLAB. In this section, we report the results of the conducted simulations. The sensors are distributed uniformly at random in a square-shaped bounded environment A of size 120×120 or 240×240 units. The number of sensors k is varied between 50 and 150, number of deployed UAVs n is varied between 5 and 10, hop threshold τ is varied between 1 and 3, each sensor initially has $c_s = 1$ units of data, and the required time to collect $c_s = 1$ units of data is $T = 6$. For altitude $h = 10$, the expansion radius r is varied between 5 and 30, thus the communication range R is varied between $\approx 11 - 31$. For $A = 240 \times 240$, $d = 340$, and for $A = 120 \times 120$, $d = 170$. In each analysis, we vary two parameters while fixing the others and take the average over 100 trials to obtain the result of the analysis. We conducted two set of simulations: Group-1 validates the performance of our algorithm, while Group-2 provides insight into the degree of aggregation and network partitioning and shows the effects of having Stage-1 on the overall performance. In Group-1, the hop threshold $\tau = 1$.

In Group-1, we use the following metrics for evaluation: (1) Distance traveled: The maximum distance traveled in Stage-2 and Stage-3 among the UAVs; (2) Total time: The longest mission completion time among the UAVs. The total time for a UAV is calculated as the sum of the distance traveled and the waiting time by the UAV both in Stage-2 and Stage-3 of CDC. The waiting time is represented in distance traveled units and includes the time the UAV waits for the UGV to be recharged both in Stage-2 and Stage-3, and the time to collect sensor data in Stage-3; (3) Waiting time in Stage-2 and Stage-3; (4) Total recharging count: The maximum number of charging services received from the UGV among the UAVs.

In Group-2, we use the following metrics for evaluation: (1) Number single sensors and networked sensors with respect to the initial configuration of the sensors; (2) Percentage of the number of single sensors to the number of CH sensors at the end of Stage-1; (3) Number of sensors with $c_s = 1$ units of data at the end of Stage-1 to be collected in Stage-3; (4) Number of single and CH sensors whose data will be collected

in Stage-3; (5) Number of CHs at the end of Stage-1; (6) Number of single and CH sensors at the end of Stage-1.

Group-1 shows the performance of the CDC algorithm for fixed n and R varying A and k , for fixed R varying A , n and k , and for fixed A and n varying R , the distance traveled and total time decreases as A decreases and approaches to a constant value over increasing k as A gets smaller (see figures 5.1 and 5.4). In Figures 5.10 and 5.11, we also observe that as A increases, the UAVs require recharging service more which increases the total waiting time (see figures 5.7 and 5.8). Figures 5.2 and 5.5 show that the distance and time efficiency increase over the increasing number of UAVs. In figures 5.3 and 5.6, we investigate the influence of communication range on total distance and time efficiency. The results show that increasing R while maintaining a constant number of UAVs and environment size improves the performance. They further demonstrate the scalability of our algorithm with respect to k which is enhanced by the clustering approach used in Stage-1.

Group-2 analyzes how combinations of k , R , and τ influence the formation of single and networked sensors (see figure 5.13), and the outcome of Stage-1 (see figures 5.14-5.18), thus the algorithm's performance. For fixed A , increasing R and k affect the formation of single and networked sensors and also shapes the number of sensors with $c_s = 1$ units of data at the end of Stage-1 to be collected in Stage-3 (see figure 5.13). In figure 5.12, single sensor to CH sensor percentage decreases as R increases also as the sensor density increases. In figure 5.17 and 5.18, we observe that increasing τ leads the number of sub-clusters of a cluster to decrease, since the size of a sub-cluster (i.e. the number sensors assigned to a CH) increases. This leads to a reduction in the total number of CHs, thus the total number of single and CH sensors. As r (thus, R for fixed h) increases, the number of CHs decreases. This outcome also causes the number of single and CH sensors whose data will be collected in Stage-3 to decrease (see figure 5.16), which in turn improves the total distance traveled and time performance of the UAVs (see figures 5.15 and 5.18). Given that A , R and k fixed, figure 5.17 shows that as c_u (i.e. n) increases, the number of CHs decreases. This is because the number of sub-clusters decreases as capacity increases (refer to Figure 4.2. Pseudocode of CDC: Stage-1, line 21), so the number of CHs that are determined for these sub-clusters also decreases.

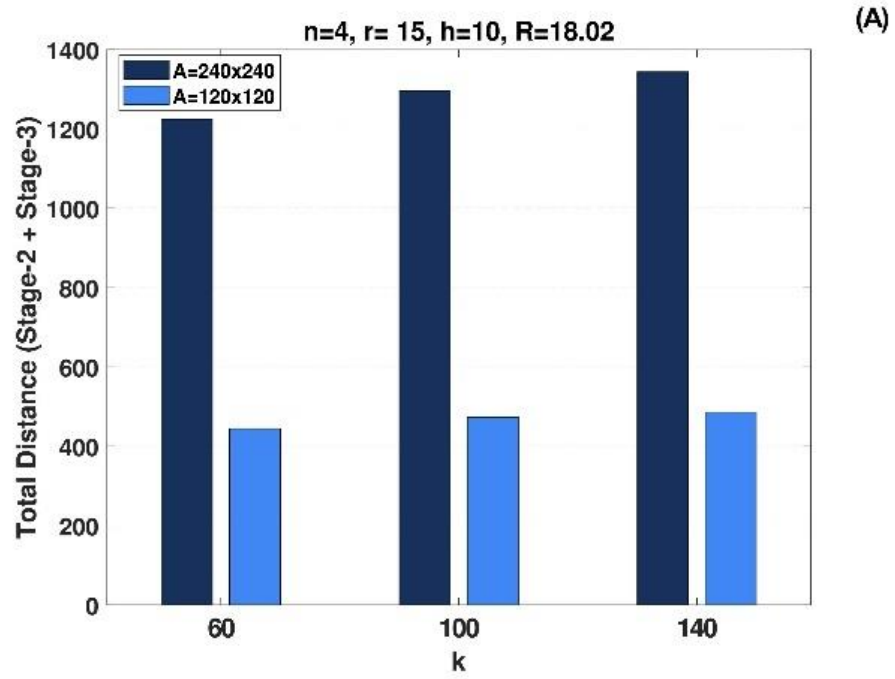


Figure 5.1 Total Distance Changes over A and k

Source: Author

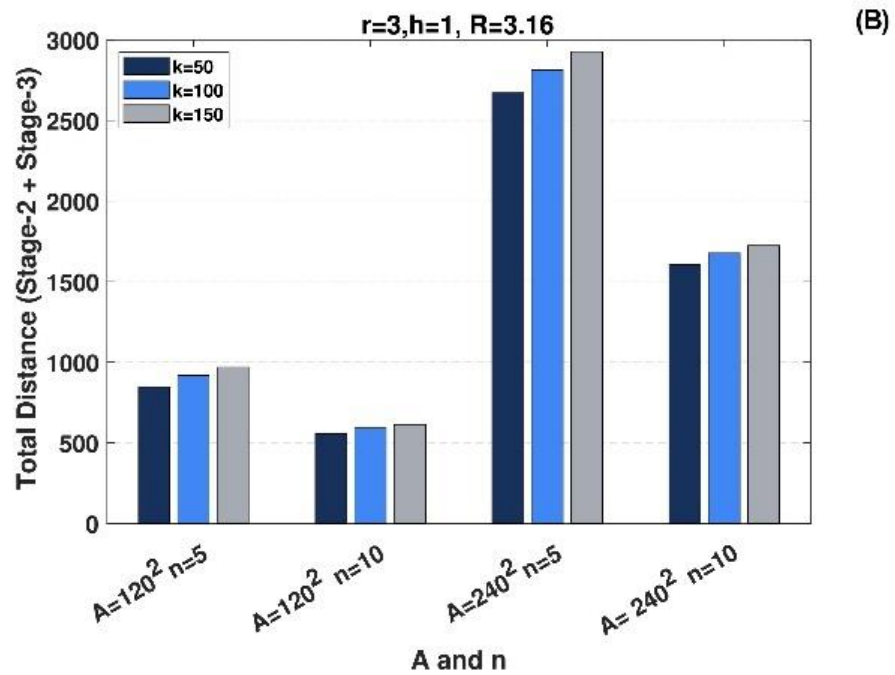


Figure 5.2 Total Distance Changes over A , n and k

Source: Author

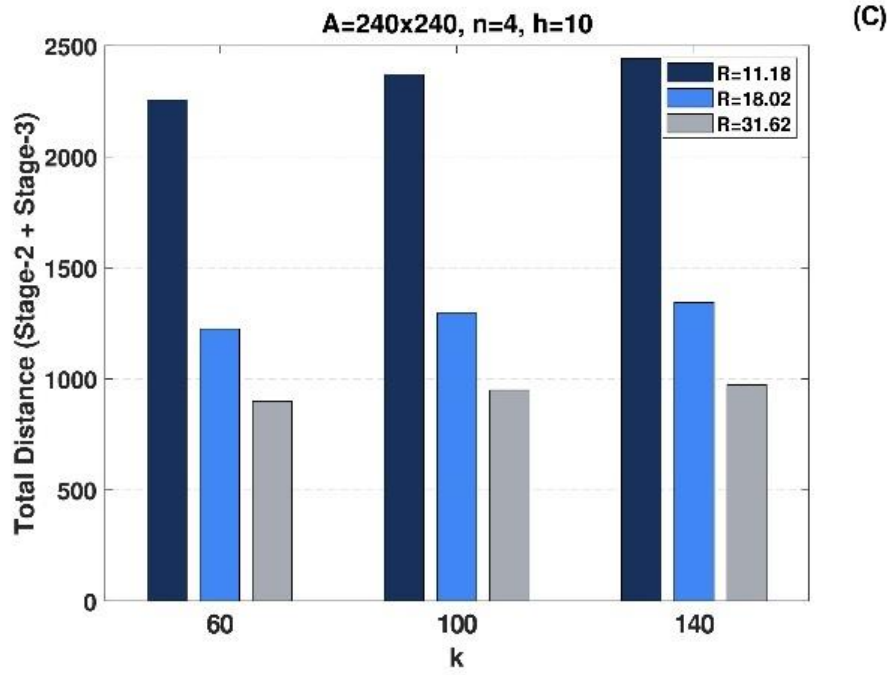


Figure 5.3 Total Distance Changes over R and k

Source: Author

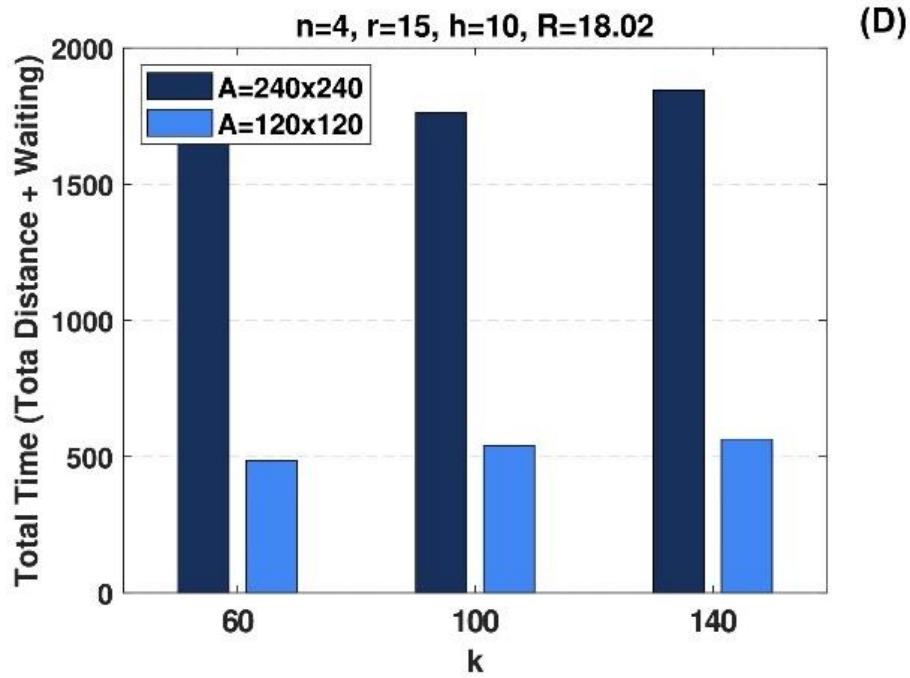


Figure 5.4 Total Time Changes over A and k

Source: Author

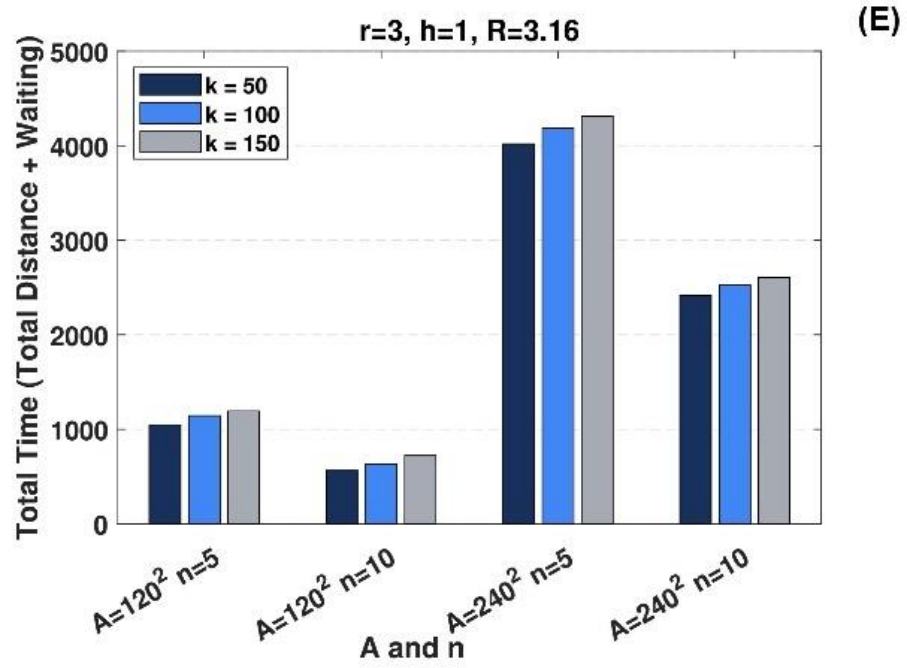


Figure 5.5 Total Time Changes over A , n and k

Source: Author

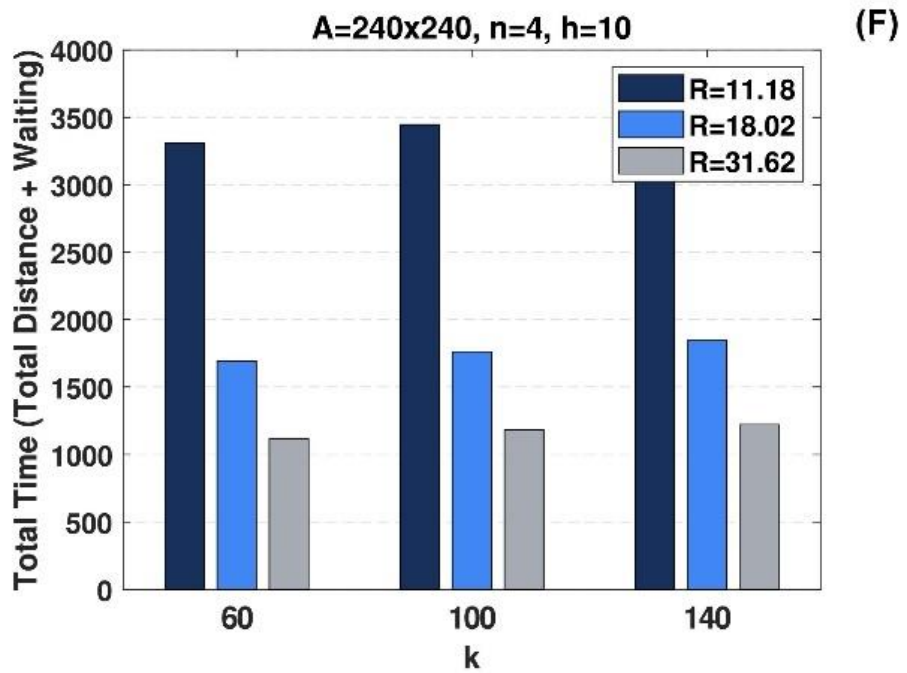


Figure 5.6 Total Time Changes over R and k

Source: Author

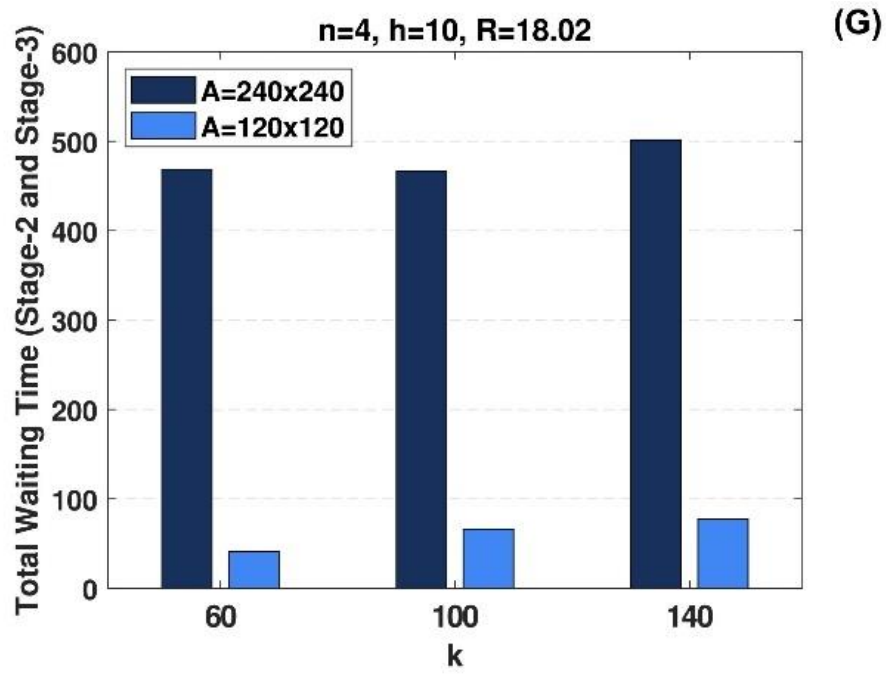


Figure 5.7 Total Waiting Time Changes over A and k

Source: Author

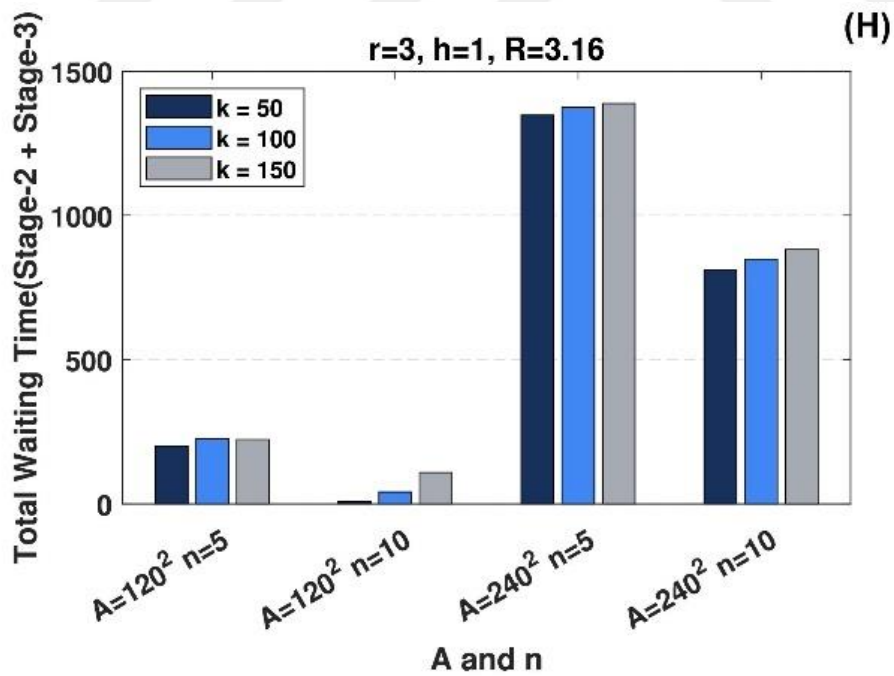


Figure 5.8 Total Waiting Time Changes over A , n and k

Source: Author

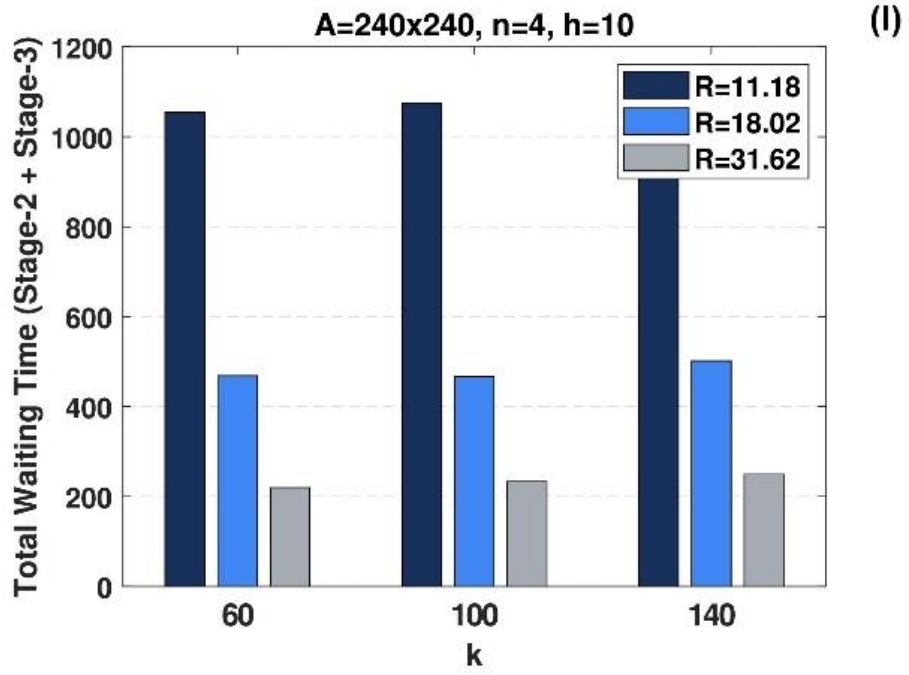


Figure 5.9 Total Waiting Time Changes over R and k

Source: Author

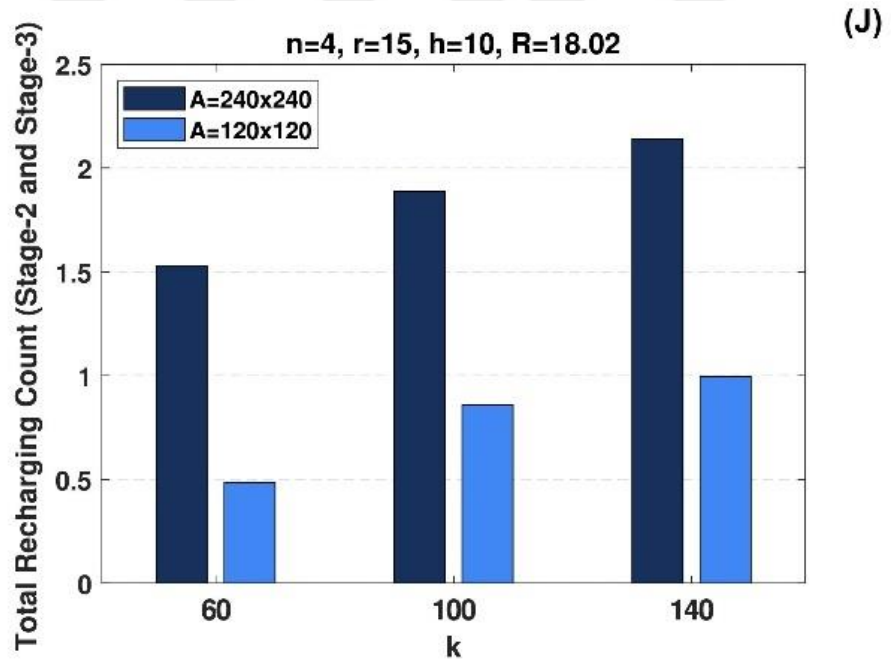


Figure 5.10 Total Recharging Count Changes over A and k

Source: Author

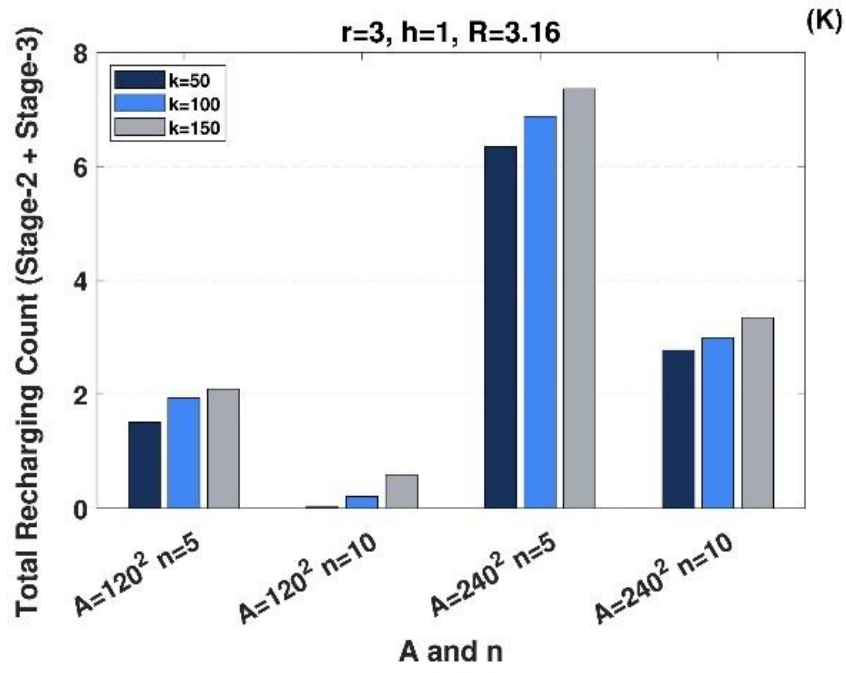


Figure 5.11 Total Recharging Count Changes over A , n and k

Source: Author

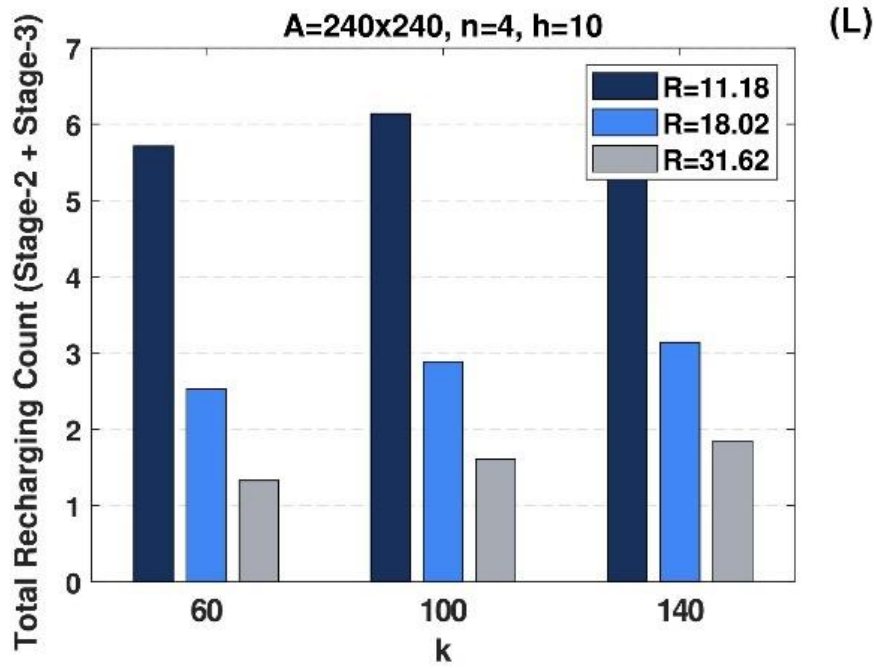


Figure 5.12 Total Recharging Count Changes over R and k

Source: Author

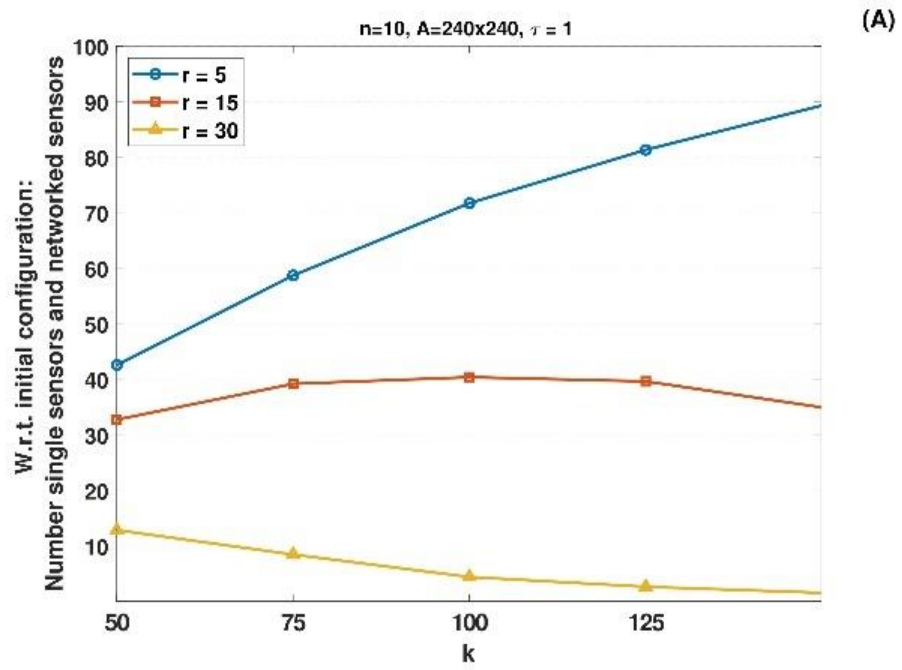


Figure 5.13 Number of Initial CHs

Source: Author

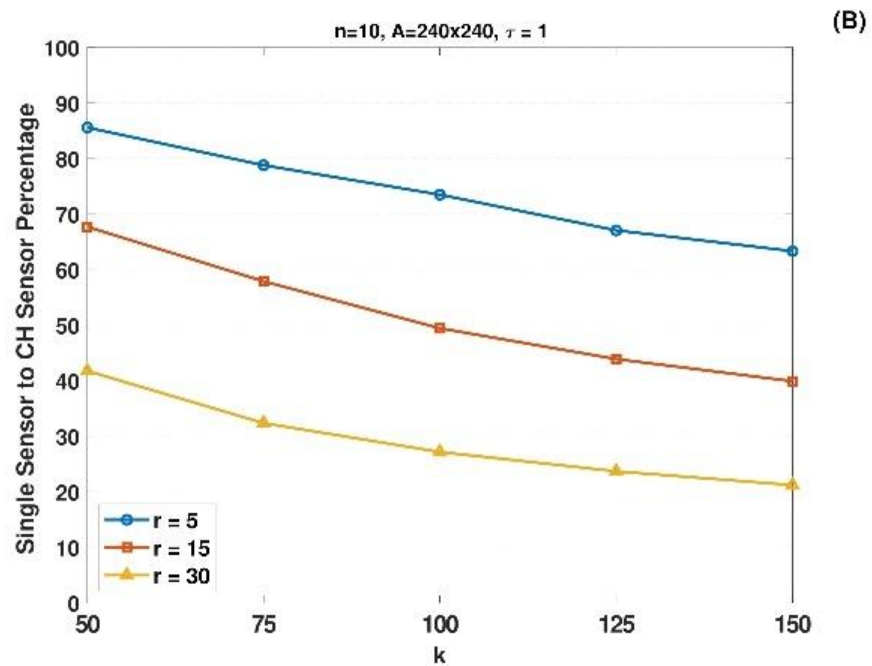


Figure 5.14 Percentage of Single Sensors over CHs

Source: Author

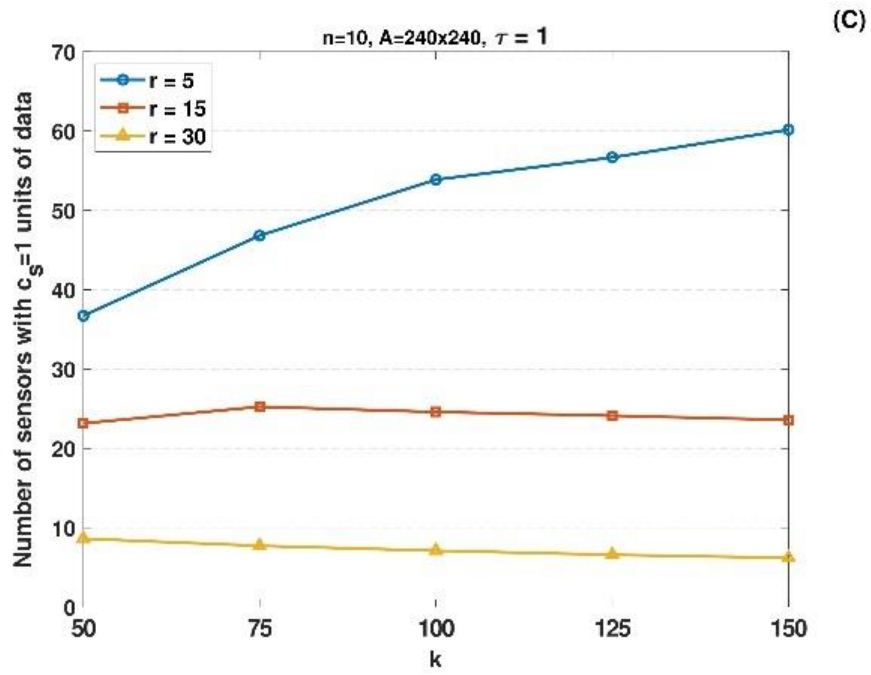


Figure 5.15 Number of Single Sensors with Demand Value of 1

Source: Author

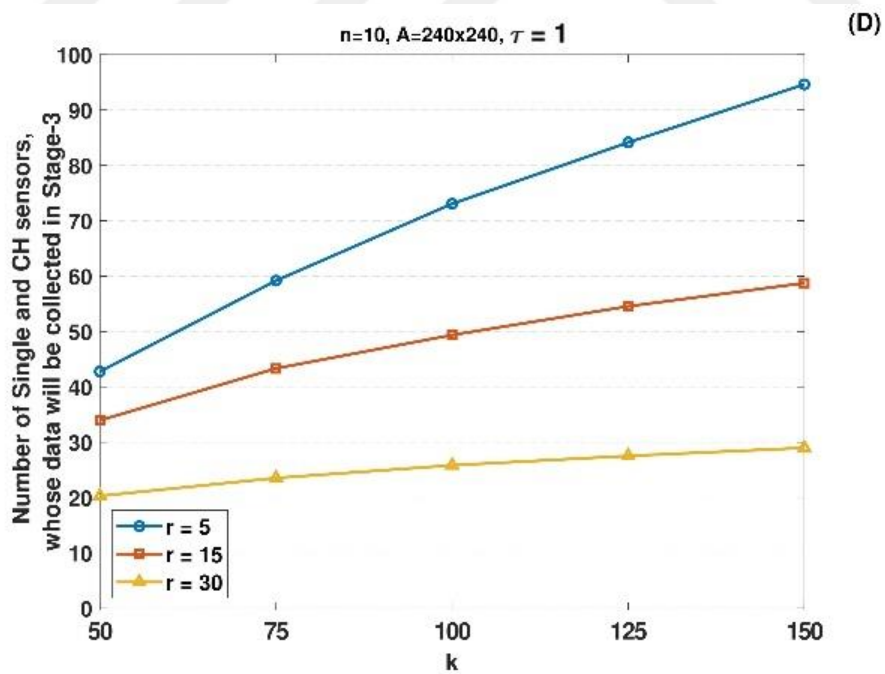


Figure 5.16 Total Number of Cluster heads

Source: Author

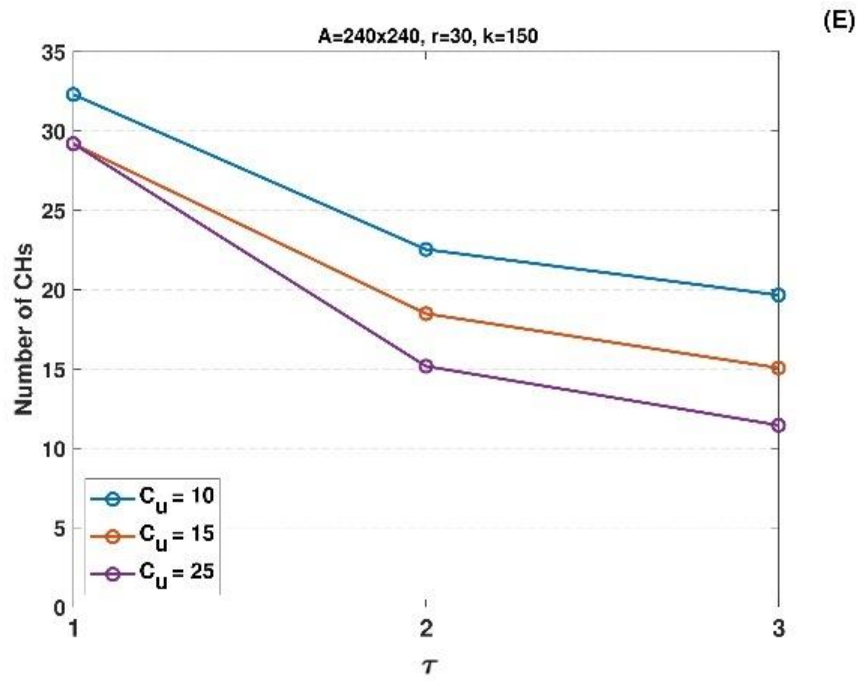


Figure 5.17 Number of Cluster heads Changes over UAV Capacities

Source: Author

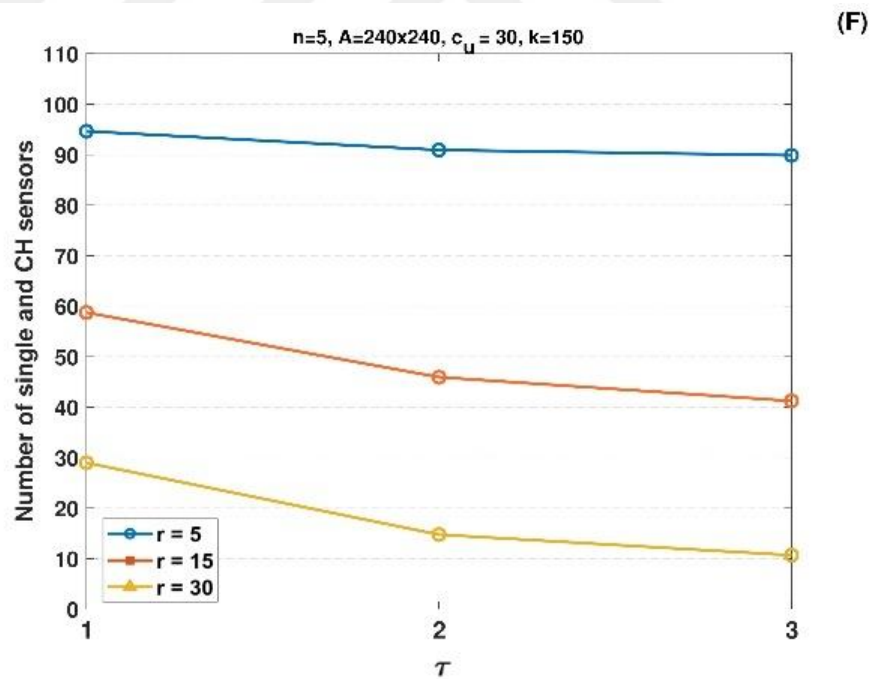


Figure 5.18 Effect of Hop Threshold Values on Number of CHs

Source: Author

6. CHAPTER: CONCLUSION

In this thesis, we investigated the problem of data collection from a set of stationary wireless sensor nodes with a team of capacity-constrained UAVs that are assisted by a UGV. The agents are operating in a large bounded area and do not know the initial locations of the sensors. Throughout the mission, the UAVs work as data-collectors and the UGV work as service-provider to the UAVs. We consider partitioned WSNs such that a sensor may be single or a member of networked sensors. For the formulated problem, we proposed a solution that plans the paths of the UAVs and UGV efficiently to minimize the overall mission time. We combined the techniques including the partitioning the environment and assigning each region to a UAV, clustering and cluster head selection and solving CVRP. We empirically evaluated the performance of our algorithm in simulations.

Application Domains - The applicability of the proposed method is particularly relevant for open areas. Agricultural fields typically consist of large, open, and relatively flat areas, making them highly suitable for the proposed approach. To support precision agriculture practices, UAV-UGV collaboration can efficiently collect data from sensors measuring parameters such as soil moisture, temperature, fertilization levels, and pest density. Following disasters such as floods in plains or rural areas, the proposed system can be employed to collect data from sensors deployed across the affected region. In cases where communication infrastructure is damaged, UAVs act as mobile data collectors while UGV-supported recharging ensures uninterrupted operation. This supports emergency response planning.

Future Work - Although we assume that UAVs and UGV move at unit speed v , our approach can be adapted for the case where the UAV and UGV speeds are different. A key direction for future work is to investigate the coordination of multiple UGVs to reduce UAV waiting times. Another interesting future work is to study the problem in complex environments, i.e. environments with obstacles. Given that the environment is unknown, the path planning both in Stage-2 and Stage-3 of the CDC algorithm

should be changed. Moreover, such environments also make determining predefined rendezvous locations challenging.



7. REFERENCES

- Abu-Baker, A., Shakhathreh, H., Sawalmeh, A., & Alenezi, A. H. (2023). Efficient Data Collection in UAV-Assisted Cluster-Based Wireless Sensor Networks for 3D Environment: Optimization Study. *Journal of Sensors*, 2023(1). <https://doi.org/10.1155/2023/9513868>
- Amer, A. A., Ahmed, R., Fahim, I. S., & Ismail, T. (2024). Energy Optimization and Trajectory Planning for Constrained Multi-UAV Data Collection in WSNs. *IEEE Access*, 12, 9047–9061. <https://doi.org/10.1109/access.2024.3353193>
- Batta, M. S., Mabed, H., Aliouat, Z., & Harous, S. (2021). A Distributed Multi-Hop Intra-Clustering Approach Based on Neighbors Two-Hop Connectivity for IoT Networks. *Sensors*, 21(3), 873. <https://doi.org/10.3390/s21030873>
- Bhadauria, D., Tekdas, O., & Isler, V. (2011). Robotic data mules for collecting data over sparse sensor fields. *Journal of Field Robotics*, 28(3), 388–404. <https://doi.org/10.1002/rob.20384>
- Bouhamed, O., Ghazzai, H., Besbes, H., & Massoud, Y. (2020). A UAV-Assisted Data Collection for Wireless Sensor Networks: Autonomous Navigation and Scheduling. *IEEE Access*, 8, 110446–110460. <https://doi.org/10.1109/access.2020.3002538>
- Caillouet, C., Giroire, F., & Razafindralambo, T. (2019). Efficient data collection and tracking with flying drones. *Ad Hoc Networks*, 89, 35–46. <https://doi.org/10.1016/j.adhoc.2019.01.011>
- Chen, M., Liang, W., & Li, Y. (2020). Data Collection Maximization for UAV-Enabled Wireless Sensor Networks. <https://doi.org/10.1109/icccn49398.2020.9209619>
- Chen, T.-C., Chen, T.-S., & Wu, P.-W. (2011). On Data Collection Using Mobile Robot in Wireless Sensor Networks. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 41(6), 1213–1224. <https://doi.org/10.1109/tsmca.2011.2157132>
- Chee-Yee Chong, & Kumar, S. P. (2003). Sensor networks: Evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8), 1247–1256. <https://doi.org/10.1109/jproc.2003.814918>

- Chiu, K.-M., & Liu, J.-S. (2011). *Robot routing using clustering-based parallel genetic algorithm with migration*. 42–49. <https://doi.org/10.1109/mfcist.2011.5949511>
- Dantzig, G., Fulkerson, R., & Johnson, S. (1954). Solution of a Large-Scale Traveling-Salesman Problem. *Journal of the Operations Research Society of America*, 2(4), 393–410. <https://www.jstor.org/stable/166695>
- Deng, C., Fu, X., Claudio, S., & Fortino, G. (2025). Low-AoI data collection for multi-UAVs-UGVs assisted large-scale IoT systems based on workload balancing. *Ad Hoc Networks*, 174, 103844. <https://doi.org/10.1016/j.adhoc.2025.103844>
- Dios, J. M., De, A., Antidio Viguria, Torres-González, A., & Ollero, A. (2017). Combining Unmanned Aerial Systems and Sensor Networks for Earth Observation. *Remote Sensing*, 9(4), 336–336. <https://doi.org/10.3390/rs9040336>
- Eksioglu, B., Vural, A. V., & Reisman, A. (2009). The vehicle routing problem: A taxonomic review. *Computers & Industrial Engineering*, 57(4), 1472–1483. <https://doi.org/10.1016/j.cie.2009.05.009>
- Feld, S., Roch, C., Gabor, T., Seidel, C., Neukart, F., Galter, I., Maurer, W., & Linnhoff-Popien, C. (2019). A Hybrid Solution Method for the Capacitated Vehicle Routing Problem Using a Quantum Annealer. *Frontiers in ICT*, 6. <https://doi.org/10.3389/fict.2019.00013>
- Gielis, J., Shankar, A., & Prorok, A. (2022). A Critical Review of Communications in Multi-robot Systems. *Current Robotics Reports*, 3(4), 213–225. <https://doi.org/10.1007/s43154-022-00090-9>
- Gong, J., Chang, T.-H., Shen, C., & Chen, X. (2018, January 9). *Flight Time Minimization of UAV for Data Collection over Wireless Sensor Networks*. <https://doi.org/10.48550/arXiv.1801.02799>
- Gul, O. M., Erkmén, A. M., & Burak Kantarci. (2022). UAV-Driven Sustainable and Quality-Aware Data Collection in Robotic Wireless Sensor Networks. *ResearchGate*. <https://doi.org/10.13140/RG.2.2.17052.16004/1>
- Howie Choset, & P. Pignon. (1998). Coverage Path Planning: The Boustrophedon Cellular Decomposition. *Springer EBooks*, 203–209. https://doi.org/10.1007/978-1-4471-1273-0_32
- Hu, Y., Zhang, F., Tian, T., Ma, D., & Shi, Z. (2022). Shortest path planning of a data mule in wireless sensor networks. *Wireless Networks*, 28(3), 1129–1145. <https://doi.org/10.1007/s11276-022-02891-4>
- Jünger, M., Reinelt, G., & Rinaldi, G. (1995). Chapter 4 The traveling salesman problem. *Handbooks in Operations Research and Management Science*, 7, 225–330. [https://doi.org/10.1016/S0927-0507\(05\)80121-5](https://doi.org/10.1016/S0927-0507(05)80121-5)

- Kandris, D., Nakas, C., Vomvas, D., & Koulouras, G. (2020). Applications of Wireless Sensor Networks: An Up-to-Date Survey. *Applied System Innovation*, 3(1), 14. <https://doi.org/10.3390/asi3010014>
- Liu, H., Chen, R., Ding, S., Jiang, Z., Liu, F., & Zhang, J. (2024). An energy efficiency routing protocol for UAV-aided WSNs data collection. *Ad Hoc Networks*, 154, 103378. <https://doi.org/10.1016/j.adhoc.2023.103378>
- Liu, J.-S., Wu, S.-Y., & Chiu, K.-M. (2013). *Path planning of a data mule in wireless sensor network using an improved implementation of clustering-based genetic algorithm*. <https://doi.org/10.1109/cica.2013.6611660>
- Luo, C., Chen, W., Li, D., Wang, Y., Du, H., Wu, L., & Wu, W. (2020). Optimizing flight trajectory of UAV for efficient data collection in wireless sensor networks. *Theoretical Computer Science*, 853, 25–42. <https://doi.org/10.1016/j.tcs.2020.05.019>
- Najjar-Ghabel, S., Farzinvas, L., & Razavi, S. N. (2020). Mobile sink-based data gathering in wireless sensor networks with obstacles using artificial intelligence algorithms. *Ad Hoc Networks*, 106, 102243. <https://doi.org/10.1016/j.adhoc.2020.102243>
- Pan, Q., & Fu, X. (2023). Fresh data collection for UAV-assisted IoTs based on proximity-remote region collaboration. *Ad Hoc Networks*, 146, 103182. <https://doi.org/10.1016/j.adhoc.2023.103182>
- Raju, M. R., Sai Krishna Mothku, Manoj Kumar Somesula, & Srilatha Chebrolu. (2024). Age and energy aware data collection scheme for urban flood monitoring in uav-assisted Wireless Sensor Networks. *Ad Hoc Networks*, 168, 103704–103704. <https://doi.org/10.1016/j.adhoc.2024.103704>
- Rezende, J. da C. V., Silva, R. I. da, & Souza, M. J. F. (2020). Gathering Big Data in Wireless Sensor Networks by Drone. *Sensors*, 20(23), 6954. <https://doi.org/10.3390/s20236954>
- Singh, H., Singh, M. B., Harsh Pratik, & Pratap, A. (2023). UAV and UGV Assisted Path Planning for Sensor Data Collection in Precision Agriculture. *2023 11th International Symposium on Electronic Systems Devices and Computing (ESDC)*, 1–6. <https://doi.org/10.1109/ESDC56251.2023.10149861>
- Sotiris Papatheodorou, Michalis Smyrnakis, Tembine Hamidou, & Tzes, A. (2018). Path Planning and Task Assignment for Data Retrieval from Wireless Sensor Nodes Relying on Game-Theoretic Learning. *2022 8th International Conference on Control, Decision and Information Technologies (CoDIT)*, 1073–1078. <https://doi.org/10.1109/codit.2018.8394924>

- Tang, H., Tang, C., Li, M., & Zhou, G. (2024). A mobile data collection method for balancing energy consumption and delay in strip-shaped wireless sensor networks with branches. *Ad Hoc Networks*, 164, 103627–103627. <https://doi.org/10.1016/j.adhoc.2024.103627>
- Tokekar, P., Hook, J. V., Mulla, D., & Isler, V. (2016). Sensor Planning for a Symbiotic UAV and UGV System for Precision Agriculture. *IEEE Transactions on Robotics*, 32(6), 1498–1511. <https://doi.org/10.1109/tro.2016.2603528>
- Tsilomitrou, O., Evangeliou, N., & Tzes, A. (2018). Mobile Robot Tour Scheduling acting as Data Mule in a Wireless Sensor Network. *2018 5th International Conference on Control, Decision and Information Technologies (CoDIT)*, 327–332. <https://doi.org/10.1109/codit.2018.8394864>
- Tsilomitrou, O., & Tzes, A. (2021). Mobile Data-Mule Optimal Path Planning for Wireless Sensor Networks. *Applied Sciences*, 12(1), 247. <https://doi.org/10.3390/app12010247>
- Wang, G., Lee, B.-S., & Ahn, J. Y. (2018). UAV-Assisted Cluster Head Election for a UAV-Based Wireless Sensor Network. *2018 IEEE 6th International Conference on Future Internet of Things and Cloud (FiCloud)*, 267–274. <https://doi.org/10.1109/FiCloud.2018.00046>
- Wang, Y., Hu, Z., Wen, X., Lu, Z., & Miao, J. (2020). Minimizing Data Collection Time With Collaborative UAVs in Wireless Sensor Networks. *IEEE Access*, 8, 98659–98669. <https://doi.org/10.1109/access.2020.2996665>
- Wu, Q., Sun, P., & Boukerche, A. (2019). Unmanned aerial vehicle-assisted energy-efficient data collection scheme for sustainable wireless sensor networks. *Computer Networks*, 165, 106927. <https://doi.org/10.1016/j.comnet.2019.106927>
- Wu, Q., Sun, P., & Azzedine Boukerche. (2018). *An Energy-efficient UAV-based Data Aggregation Protocol in Wireless Sensor Networks*. <https://doi.org/10.1145/3272036.3272047>
- Yick, J., Mukherjee, B., & Ghosal, D. (2008). Wireless sensor network survey. *Computer Networks*, 52(12), 2292–2330. <https://doi.org/10.1016/j.comnet.2008.04.002>

APPENDICES

7.1. APPENDIX 1. CLUSTERING APPROACH IN STAGE-3 OF THE ALGORITHM

General CVRP clustering part of the code.

```
%% 4. Cluster Initialization: Farthest-first + Nearest-to-mean
splitOccurred = false;
unassigned = 1:M;
clusters = struct('Customers',{}, 'Center',{}, 'Demand',{});
fprintf('Starting clustering: %d sensors, %d vehicles\n', M,
numVehicles);
for v = 1:numVehicles
    capThis = vehicleCapacities(v);
    fprintf('\n--- Vehicle %d (Drone %d): capacity = %d ---\n', v,
originalDroneIDs(v), capThis);
    if isempty(unassigned)
        fprintf(' No sensors left to assign.\n');
        clusters(v) = struct('Customers',[], 'Center',[],
'Demand',0);
        continue;
    end
    % 1a) Seed with farthest-from-depot
    d0 = vecnorm(customerPositions(unassigned,:) - depot,2,2);
    [~, idx0] = max(d0);
    seed = unassigned(idx0);
    origSeed = allclusterheads(seed);
    fprintf(' Seed sensor (global ID %d) (distance %.2f)\n',
origSeed, d0(idx0));

    currentCluster.Customers = seed;
    currentCluster.Center = customerPositions(seed,:);
    currentCluster.Demand = demand(seed);
    fprintf(' Initialized cluster: Demand=%d, Center=[%.2f,
%.2f]\n', currentCluster.Demand, currentCluster.Center);
    unassigned(idx0) = [];

    % 1b) Grow loop
    iter = 0;
    while true
        iter = iter + 1;
        remCap = capThis - currentCluster.Demand;
        fitMask = demand(unassigned) <= remCap;
        if ~any(fitMask)
            fprintf(' No fitting sensors remain (remCap=%d).
Stopping growth.\n', remCap);
            break;
        end
    end
end
```

```

        cands = unassigned(fitMask);
        dC = vecnorm(customerPositions(cands,:) -
currentCluster.Center,2,2);
        [~, i1] = min(dC);
        cand = cands(i1);
        origCand = allclusterheads(cand);
        fprintf('    Iter %d: fitting sensor (global ID %d)
(distance %.2f)\n', iter, origCand, dC(i1));

        currentCluster = addToCluster(currentCluster, cand,
customerPositions, demand);
        fprintf('    Updated cluster: Demand=%d, Center=[%.2f,
%.2f]\n', currentCluster.Demand, currentCluster.Center);
        unassigned(unassigned==cand) = [];
    end
    if max(allclusterheads) <= numel(uncollectedIDs) &&
min(allclusterheads) >= 1
        globalIDs_in_this_cluster =
uncollectedIDs(allclusterheads(currentCluster.Customers));
    else
        globalIDs_in_this_cluster =
allclusterheads(currentCluster.Customers);
    end
    fprintf('  Final cluster size: %d sensors (global IDs): ',
numel(currentCluster.Customers));
    fprintf('%d ', globalIDs_in_this_cluster);
    fprintf('\n');
    clusters(v) = currentCluster;

end

```

7.2. APPENDIX 2. CLUSTER HEAD DATA SPLITTING

In case of a need for data splitting of any clusterhead to multiple UAVs. This part of the code works.

```
% Handle unassigned (splitting demands) -- remains as in your code
if ~isempty(unassigned)
    fprintf('\nSplitting unassigned sensors across multiple drones
if needed...\n');
    totalRemDemand = sum(demand(unassigned));
    remainingCaps = vehicleCapacities(:) - [clusters.Demand]';

    if sum(remainingCaps) < totalRemDemand
        error('Not enough total drone capacity to handle unassigned
clusterheads even after splitting.');
```

```
    end

    virtualCustomerPositions = zeros(0, 2);
    virtualDemands = zeros(0, 1);
    newIDs = [];
    baseVirtualID = 1000;

    for idx = unassigned
        d_remain = demand(idx);
        coord = customerPositions(idx,:);
        origID = allclusterheads(idx);
        fprintf(' Splitting sensor (global ID %d, demand=%d)\n',
origID, d_remain);

        for v = 1:numVehicles
            cap = remainingCaps(v);
            if cap <= 0, continue; end
            d_use = min(cap, d_remain);
            if d_use <= 0, continue; end

            virtualCustomerPositions = [virtualCustomerPositions;
coord];

            virtualDemands = [virtualDemands; d_use];
            virtualID = baseVirtualID + size(virtualDemands,1);
            newIDs(end+1) = virtualID;

            clusters(v).Customers(end+1) =
size(customerPositions,1) + size(virtualCustomerPositions,1);
            clusters(v).Demand = clusters(v).Demand + d_use;
            remainingCaps(v) = remainingCaps(v) - d_use;
            d_remain = d_remain - d_use;
            if d_remain <= 0, break; end
        end
    end
    if d_remain > 0
```

```

        error('Failed to fully assign split demand of sensor
%d', origID);
    end
end
    customerPositions = [customerPositions;
virtualCustomerPositions];
    demand = [demand; virtualDemands];
    allclusterheads = [allclusterheads(:); newIDs(:)];
    fprintf('All remaining demands split successfully.\n');
    splitOccurred = true;
end

```

7.3. APPENDIX 3. BOUSTROPHEDON PATH GENERATION

Boustrophedon path generation and motion planning phase.

```

plane_length = evalin('base','plane_length');
y_length = evalin('base','y_length');
x_min = -plane_length/2; x_max = plane_length/2;
y_min = -y_length/2; y_max = y_min + y_length;

num_sensors = length(sensor_x);
sensor_capacity_per_drone = evalin('base','sensor_capacity_of_drones');
partition_length = plane_length / num_drones;

battery_capacity = evalin('base','batteryCapacity');
consumption_rate = 1;
droneBattery = battery_capacity * ones(1, num_drones);
step_resolution = 1;

recharger_pos = [0, 0];
recharger_speed = step_resolution;
totalRechargerDist = 0;
distAtRequest = nan(1, num_drones);

rechargerWaitTotal = zeros(1, num_drones);
rechargeQueue = [];
serviceCount = zeros(1, num_drones);
currentService = [];
rechargerMemory = [];
stopSignalSent = false;
sensorIDs = (1:num_sensors)';
found_sensors = false(1,num_sensors);
collected_sensors= false(1,num_sensors);

step_size = evalin('base','stripe_size');

```

```

sample_resolution=1;
collider_half=evalin('base','expansion_radius');
drone_start_x = x_min + collider_half + (0:num_drones-
1)*partition_length;
drone_start_y = repmat(y_max - collider_half, 1, num_drones);

drone_distances      = zeros(1,num_drones);
drone_collected_count= zeros(1,num_drones);
currentDronePositions= [drone_start_x; drone_start_y]';

droneHaltTimes      = zeros(1,num_drones);
droneHaltStart      = nan(1,num_drones);
recharge_duration= 0;
droneRechargeStart  = nan(1,num_drones);

dronePaths = cell(1,num_drones);
for i=1:num_drones
    part_x_min = x_min + (i-1)*partition_length;
    part_x_max = part_x_min + partition_length;
    xs = part_x_min+collider_half:step_size:part_x_max-collider_half;
    cy = y_max-collider_half; path=[];
    for j=1:length(xs)
        tx = xs(j);
        if mod(j,2) == 1
            ty = y_min + collider_half;
        else
            ty = y_max - collider_half;
        end
        n = ceil(abs(ty-cy)/sample_resolution)+1;
        path = [path; [repmat(tx,n,1), linspace(cy,ty,n)']];
        cy = ty;
        if j<length(xs)
            nx = xs(j+1);
            n2=ceil(abs(nx-tx)/sample_resolution)+1;
            path=[path; [linspace(tx,nx,n2)', repmat(cy,n2,1)]];
        end
    end
    dronePaths{i}=path;
end

```

7.4. APPENDIX 4. FOLLOWING THE BOUSTROPHEDON PATH

```

% === Boustrophedon MAIN SIMULATION LOOP ===
while ~( ~stopSignalGiven && all(droneStep > cellfun(@(p)
size(p,1),dronePaths)) )
    haltedDroneInfo = [];
    elapsedTime = toc(mission_start);

```

```

    for i=1:num_drones
        if droneBattery(i)>0 && droneStep(i)<=size(dronePaths{i},1)
            pos = dronePaths{i}(droneStep(i),:);
            d = norm(pos-previousPositions(i,:));
            drone_distances(i) = drone_distances(i)+d;
            droneBattery(i) = max(droneBattery(i)-
consumption_rate*d,0);
            previousPositions(i,:)=pos;
            currentDronePositions(i,:)=pos;

            for s=1:num_sensors
                if ~found_sensors(s) && abs(sensor_x(s)-
pos(1))<=collider_half ...
                                && abs(sensor_y(s)-
pos(2))<=collider_half
                    found_sensors(s) = true;
                    perDroneCollected{i}(end+1) = s;
                end
            end

            if droneBattery(i) > 0
                droneStep(i) = droneStep(i) + 1;
            end
        end
        if droneBattery(i)==0
            if isnan(droneHaltStart(i)), droneHaltStart(i)=elapsedTime;
end
            haltedDroneInfo(end+1,:)=[i,currentDronePositions(i,:)];
        end
    end
end

```

7.5. APPENDIX 5. EXAMPLE - OUTPUT OF STAGE-2

Example output screenshot for Stage-2. The number of sensors is $k=60$ and the number of UAVs is $n=4$. As in the system, the size of each stripe is the same. Therefore, all UAVs have the same number of recharging rounds. The distance traveled by each UAV is the same. And, in this stage, the UAVs do not collect data from any sensor.

Command Window

```
>> testtttv2
1

--- Drone başına Recharger Toplam Bekleme Mesafesi ---
Drone 1: Recharger beklerken TOPLAM 672.81 birim hareket etmiş.
Drone 2: Recharger beklerken TOPLAM 732.81 birim hareket etmiş.
Drone 3: Recharger beklerken TOPLAM 792.81 birim hareket etmiş.
Drone 4: Recharger beklerken TOPLAM 853.52 birim hareket etmiş.
Total Mission Time: 0.11 seconds

--- Recharger Toplam Mesafe & Şarj Sayısı ---
Drone 1: Toplam Mesafe = 672.81 birim, Şarj Sayısı = 4
Drone 2: Toplam Mesafe = 732.81 birim, Şarj Sayısı = 4
Drone 3: Toplam Mesafe = 792.81 birim, Şarj Sayısı = 4
Drone 4: Toplam Mesafe = 853.52 birim, Şarj Sayısı = 4

--- Halt Time Summary ---
Drone 1 was halted for a total of 0.04 seconds
Drone 2 was halted for a total of 0.05 seconds
Drone 3 was halted for a total of 0.06 seconds
Drone 4 was halted for a total of 0.06 seconds
--- Sensor Collection Summary ---
Drone 1 collected 0/15 sensors
Drone 2 collected 0/15 sensors
Drone 3 collected 0/15 sensors
Drone 4 collected 0/15 sensors

--- Distance Traveled Summary ---
Drone 1 traveled 1430.00 units
Drone 2 traveled 1430.00 units
Drone 3 traveled 1430.00 units
Drone 4 traveled 1430.00 units
```

7.6. APPENDIX 6. EXAMPLE - LIST OF THE FOUND SENSORS DURING STAGE-2

The list of sensors that the UAVs found in Stage-1, i.e. during the exploration of the environment.

```
Command Window
Maximum distance traveled: 1430.00 units by drone(s): 1 2 3 4
✔ Drone distances saved to "drone_distances.xlsx"
--- Non-Collected Sensors ---
Sensor 1: Position = [3.16, -9.86]
Sensor 2: Position = [-86.67, -107.70]
Sensor 3: Position = [29.97, -104.37]
Sensor 4: Position = [-90.94, 114.21]
Sensor 5: Position = [82.56, 91.66]
Sensor 6: Position = [12.61, 81.75]
Sensor 7: Position = [62.13, 96.89]
Sensor 8: Position = [52.83, -30.35]
Sensor 9: Position = [-84.03, 52.63]
Sensor 10: Position = [37.14, 116.01]
Sensor 11: Position = [11.57, -117.92]
Sensor 12: Position = [-109.44, -11.45]
Sensor 13: Position = [-55.20, -49.42]
Sensor 14: Position = [10.14, 27.18]
Sensor 15: Position = [-57.83, -13.54]
Sensor 16: Position = [-8.94, 91.43]
Sensor 17: Position = [-2.01, -29.69]
Sensor 18: Position = [36.57, 51.86]
Sensor 19: Position = [-31.91, -6.46]
Sensor 20: Position = [-15.31, -54.50]
Sensor 21: Position = [-18.71, 44.01]
Sensor 22: Position = [63.92, -41.33]
Sensor 23: Position = [110.33, 88.47]
Sensor 24: Position = [49.62, 56.24]
Sensor 25: Position = [62.65, 21.36]
Sensor 26: Position = [68.49, 44.24]
Sensor 27: Position = [-91.85, 86.75]
Sensor 28: Position = [85.35, -107.85]
Sensor 29: Position = [46.42, -108.72]
Sensor 30: Position = [118.11, 96.38]
Sensor 31: Position = [91.28, -102.76]
Sensor 32: Position = [89.86, -30.22]
Sensor 33: Position = [109.78, -92.78]
Sensor 34: Position = [-36.72, 46.21]
Sensor 35: Position = [-37.36, 102.13]
Sensor 36: Position = [92.31, -52.40]
Sensor 37: Position = [0.83, 17.48]
Sensor 38: Position = [76.23, -25.53]
Sensor 39: Position = [82.52, -10.83]
Sensor 40: Position = [25.32, 92.73]
Sensor 41: Position = [93.13, 89.52]
Sensor 42: Position = [-93.07, 102.68]
Sensor 43: Position = [86.68, -34.51]
Sensor 44: Position = [-71.43, 85.68]
Sensor 45: Position = [-14.99, -106.34]
Sensor 46: Position = [117.52, -102.32]
Sensor 47: Position = [38.98, 88.88]
Sensor 48: Position = [-74.62, 87.82]
Sensor 49: Position = [115.23, 34.28]
Sensor 50: Position = [12.11, 34.94]
Sensor 51: Position = [-12.93, -35.67]
Sensor 52: Position = [44.20, -103.99]
Sensor 53: Position = [3.56, -42.60]
Sensor 54: Position = [62.05, 110.40]
Sensor 55: Position = [52.65, 75.64]
Sensor 56: Position = [35.23, -5.79]
Sensor 57: Position = [-98.87, 113.70]
Sensor 58: Position = [21.14, -66.78]
Sensor 59: Position = [-51.55, -70.86]
Sensor 60: Position = [-50.51, -86.59]
```

7.7. APPENDIX 7. EXAMPLE – THE SET OF CLUSTERS AFTER STAGE-1

Clustering of the sensors in Stage-1 with respect to the communication range between each other. There are $k=60$ sensors in the environment. Here, we see the outputs $\bar{S}$ and $\bar{k}$. 16 of the sensors are clustered and 8 clusters have a capacity greater than 1. We have $\bar{k} = 52$.

```
Command Window

clusters =

52×1 cell array

{[ 1]}
{[ 2]}
{[ 3]}
{[ 4 57]}
{[ 5 41]}
{[ 6]}
{[ 7]}
{[ 8]}
{[ 9]}
{[ 10]}
{[ 11]}
{[ 12]}
{[ 13]}
{[14 50]}
{[ 15]}
{[ 16]}
{[ 17]}
{[ 18]}
{[ 19]}
{[ 20]}
{[ 21]}
{[ 22]}
{[23 30]}
{[ 24]}
{[ 25]}
{[ 26]}
{[ 27]}
{[28 31]}
{[29 52]}
{[32 43]}
{[ 33]}
{[ 34]}
{[ 35]}
{[ 36]}
{[ 37]}
{[ 38]}
{[ 39]}
{[ 40]}
{[ 42]}
{[44 48]}
{[ 45]}
{[ 46]}
{[ 47]}
{[ 49]}
{[ 51]}
{[ 53]}
{[ 54]}
{[ 55]}
{[ 56]}
{[ 58]}
{[ 59]}
{[ 60]}
```

7.8. APPENDIX 8. EXAMPLE - CVRP ROUTE GENERATION PHASE

Since the number of cluster heads at the end of Stage-1 is $\bar{k} = 52$, the routes in Stage-3 are determined according to the positions of these 52 CHs. The screenshot below shows the assigned sensors on the routes of only the first two UAVs.

```
Command Window
Initial cluster (connected component) sayısı: 52

--- Subcluster/Clusterhead Statistics ---
Toplam clusterhead (subcluster) sayısı      : 52
Yalnızca 1 sensör toplayan clusterhead sayısı: 44
Yüzde olarak                               : 84.62%
Starting clustering: 52 sensors, 4 vehicles

--- Vehicle 1 (Drone 1): capacity = 15 ---
Seed sensor (global ID 46) (distance 155.82)
  Initialized cluster: Demand=1, Center=[117.52, -102.32]
  Iter 1: fitting sensor (global ID 33) (distance 12.29)
    Updated cluster: Demand=2, Center=[113.65, -97.55]
  Iter 2: fitting sensor (global ID 28) (distance 30.12)
    Updated cluster: Demand=4, Center=[104.22, -100.98]
  Iter 3: fitting sensor (global ID 36) (distance 50.02)
    Updated cluster: Demand=5, Center=[101.24, -88.84]
  Iter 4: fitting sensor (global ID 29) (distance 58.32)
    Updated cluster: Demand=7, Center=[90.28, -92.81]
  Iter 5: fitting sensor (global ID 22) (distance 57.84)
    Updated cluster: Demand=8, Center=[85.88, -84.23]
  Iter 6: fitting sensor (global ID 32) (distance 54.16)
    Updated cluster: Demand=10, Center=[86.45, -76.52]
  Iter 7: fitting sensor (global ID 38) (distance 52.00)
    Updated cluster: Demand=11, Center=[85.17, -70.14]
  Iter 8: fitting sensor (global ID 8) (distance 51.27)
    Updated cluster: Demand=12, Center=[81.58, -65.72]
  Iter 9: fitting sensor (global ID 39) (distance 54.90)
    Updated cluster: Demand=13, Center=[81.67, -60.23]
  Iter 10: fitting sensor (global ID 58) (distance 60.89)
    Updated cluster: Demand=14, Center=[76.17, -60.83]
  Iter 11: fitting sensor (global ID 3) (distance 63.48)
    Updated cluster: Demand=15, Center=[72.32, -64.46]
  No fitting sensors remain (remCap=0). Stopping growth.
Final cluster size: 12 sensors (global IDs): 46 33 28 36 29 22 32 38 8 39 58 3

--- Vehicle 2 (Drone 2): capacity = 15 ---
Seed sensor (global ID 4) (distance 145.99)
  Initialized cluster: Demand=2, Center=[-90.94, 114.21]
  Iter 1: fitting sensor (global ID 42) (distance 11.72)
    Updated cluster: Demand=3, Center=[-92.00, 108.44]
  Iter 2: fitting sensor (global ID 27) (distance 21.70)
    Updated cluster: Demand=4, Center=[-91.95, 101.21]
  Iter 3: fitting sensor (global ID 44) (distance 25.74)
    Updated cluster: Demand=6, Center=[-86.82, 97.33]
  Iter 4: fitting sensor (global ID 9) (distance 44.78)
    Updated cluster: Demand=7, Center=[-86.26, 88.39]
  Iter 5: fitting sensor (global ID 35) (distance 50.80)
    Updated cluster: Demand=8, Center=[-78.11, 90.68]
  Iter 6: fitting sensor (global ID 34) (distance 60.76)
    Updated cluster: Demand=9, Center=[-72.20, 84.33]
  Iter 7: fitting sensor (global ID 16) (distance 63.66)
    Updated cluster: Demand=10, Center=[-64.29, 85.21]
  Iter 8: fitting sensor (global ID 21) (distance 61.44)
    Updated cluster: Demand=11, Center=[-59.23, 80.64]
  Iter 9: fitting sensor (global ID 6) (distance 71.84)
    Updated cluster: Demand=12, Center=[-52.04, 80.75]
  Iter 10: fitting sensor (global ID 40) (distance 78.29)
    Updated cluster: Demand=13, Center=[-45.01, 81.84]
  Iter 11: fitting sensor (global ID 14) (distance 77.65)
    Updated cluster: Demand=15, Center=[-40.41, 77.28]
  No fitting sensors remain (remCap=0). Stopping growth.
Final cluster size: 12 sensors (global IDs): 4 42 27 44 9 35 34 16 21 6 40 14
```

7.9. APPENDIX 9. EXAMPLE - THE ROUTES AND DISTANCE TRAVELED BY THE UAVS IN STAGE-3

After assigning the CHs to the UAVs. The output from the execution of Stage-3 of our algorithm. The data below shows the order of the sensors visited by each UAV in its route, the total distance traveled by the UAVs and the UGV, and the total number of recharging rounds.

```
Command Window

Final Route Assignments (demand per drone):
Drone 1: 15 sensors (global IDs): 46 33 28 36 29 22 32 38 8 39 58 3
Drone 2: 15 sensors (global IDs): 4 42 27 44 9 35 34 16 21 6 40 14
Drone 3: 15 sensors (global IDs): 23 5 7 54 55 47 10 24 18 26 25 49 56
Drone 4: 15 sensors (global IDs): 2 60 59 13 20 45 51 53 17 15 19 1 37 11 12

--- CVRP Route Distance Summary from 0.0 ---
Drone 1 covers 409.12 units
Drone 2 covers 457.04 units
Drone 3 covers 475.18 units
Drone 4 covers 539.28 units

--- CVRP Routes (with original sensor IDs) ---
Drone 1 route: Depot -> 8 -> 22 -> 38 -> 39 -> 32 -> 36 -> 33 -> 46 -> 28 -> 29 -> 3 -> 58 -> Depot
Drone 2 route: Depot -> 9 -> 27 -> 42 -> 4 -> 44 -> 35 -> 16 -> 40 -> 6 -> 34 -> 21 -> 14 -> Depot
Drone 3 route: Depot -> 18 -> 55 -> 47 -> 10 -> 7 -> 54 -> 5 -> 23 -> 49 -> 26 -> 24 -> 25 -> 56 -> Depot
Drone 4 route: Depot -> 1 -> 17 -> 51 -> 53 -> 20 -> 11 -> 45 -> 60 -> 2 -> 59 -> 13 -> 12 -> 15 -> 19 -> 37 -> Depot

Final Route Assignments (demand per drone):
Drone 1: 15 sensors (global IDs): 46 33 28 36 29 22 32 38 8 39 58 3
Drone 2: 15 sensors (global IDs): 4 42 27 44 9 35 34 16 21 6 40 14
Drone 3: 15 sensors (global IDs): 23 5 7 54 55 47 10 24 18 26 25 49 56
Drone 4: 15 sensors (global IDs): 2 60 59 13 20 45 51 53 17 15 19 1 37 11 12

--- CVRP Route Distance Summary from halted positions---
Drone 1 covers 640.82 units
Drone 2 covers 668.58 units
Drone 3 covers 569.16 units
Drone 4 covers 863.58 units

--- CVRP Routes (with original sensor IDs) ---
Drone 1 route: (-65.00, 115.00) -> 8 -> 22 -> 38 -> 39 -> 32 -> 36 -> 33 -> 46 -> 28 -> 29 -> 3 -> 58 -> (-65.00, 115.00)
Drone 2 route: (-5.00, 115.00) -> 40 -> 6 -> 16 -> 35 -> 44 -> 21 -> 14 -> 34 -> 9 -> 27 -> 42 -> 4 -> (-5.00, 115.00)
Drone 3 route: (55.00, 115.00) -> 10 -> 7 -> 47 -> 55 -> 18 -> 24 -> 56 -> 25 -> 26 -> 49 -> 5 -> 23 -> 54 -> (55.00, 115.00)
Drone 4 route: (115.00, 115.00) -> 53 -> 51 -> 20 -> 11 -> 45 -> 60 -> 2 -> 59 -> 13 -> 12 -> 15 -> 19 -> 17 -> 1 -> 37 -> (115.00, 115.00)
excel writing
animation başladi

Re-charger robotunun TOPLAM katettiği mesafe: 440.58 birim

=== Drone Halt & Recharge Report ===
Drone 1 halted 0.01 seconds and was recharged 2 times.
Drone 2 halted 0.01 seconds and was recharged 1 times.
Drone 3 halted 0.01 seconds and was recharged 1 times.
Drone 4 halted 0.01 seconds and was recharged 2 times.

=== Recharger Wait Distance Per Drone ===
Drone 1: Recharger traveled 74.49 units for this drone's recharges.
Drone 2: Recharger traveled 15.99 units for this drone's recharges.
Drone 3: Recharger traveled 31.79 units for this drone's recharges.
Drone 4: Recharger traveled 98.78 units for this drone's recharges.
=== RECHARGER LOG ===
Step 1: (10.14, 27.18) -> Drone 2, Charge 1, Time 0.02s, Mesafe: 15.99
Step 2: (68.49, 44.24) -> Drone 3, Charge 1, Time 0.03s, Mesafe: 31.79
Step 3: (92.31, -52.40) -> Drone 1, Charge 1, Time 0.03s, Mesafe: 49.47
Step 4: (11.57, -117.92) -> Drone 4, Charge 1, Time 0.03s, Mesafe: 51.02
Step 5: (21.14, -66.78) -> Drone 1, Charge 2, Time 0.04s, Mesafe: 74.49
Step 6: (-57.83, -13.54) -> Drone 4, Charge 2, Time 0.05s, Mesafe: 98.78
excel write finished
1
```