ITC 3/53
Information Technology and Control
Vol. 53 / No. 3/ 2024
pp. 865-887
DOI 10.5755/j01.itc.53.3.34530

An Efficient Framework for Sensor Data Collection by UAV Based on Clustering, Two-Fold Ant Colony Optimization and Node Grouping

Received 2023/07/01 | Accepted after revision 2024/01/18

HOW TO CITE: Shayboub, M., Al-Mahdi, E. R. H., Nassar, H. (2024). An Efficient Framework for Sensor Data Collection by UAV Based on Clustering, Two-Fold Ant Colony Optimization and Node Grouping. *Information Technology and Control*, 53(3), 865-887. https://doi.org/10.5755/j01.itc.53.3.34530

An Efficient Framework for Sensor Data Collection by UAV Based on Clustering, Two-Fold Ant Colony Optimization and Node Grouping

Magdy Shayboub, Eman Reda Hassan Al-Mahdi and Hamed Nassar

Suez Canal University, Faculty of Computers and Informatics, Computer Science Department, Ismailia 41522, Egypt; e-mails: {dr.magdy_ali, eman, drhassanwwesf, nassar}@ci.suez.edu.eg

Corresponding author: drhassanwesf@ci.suez.edu.eg

Unmanned Aerial Vehicles (UAVs) are a promising solution for sensor data collection (DC) in large-scale area. The challenge is to minimize the DC route, which will reduce UAV energy consumption and data latency. The novelty of this paper lies in its innovative approach to optimizing sensor data collection by UAVs. It combines Ant Colony Optimization (ACO) and K-means algorithms to establish an initial shortest route and introduces a unique method for grouping sensor nodes (SNs) along the route based on the UAV's footprint, reducing data latency and energy consumption for both UAV and sensors. First, an initial shortest route that traverses all SNs is established based on the ACO and the K-means algorithms. Second, we group the sensor nodes (SNs) along the initial route using the footprint of the UAV, so that the latter can collect the data of the group in one stop, instead of stopping at each SN. By sequencing the hovering locations, we obtain a (shorter) intermediate route. Third, we shorten this route even further, by applying ACO to the set of hovering locations of the intermediate route. The solution has been implemented fully in Python. The results show that the route length gets shorter progressively with each phase. To evaluate the performance of the solution objectively, we have compared it with four states of the art solutions. The results show vividly that the proposed solution produces a DC route 19.28% shorter than the shortest route produced by the four competitive solutions. Moreover, it demonstrates a remarkable improvement by retaining 44% of energy in most SNs while over 99% energy depletion observed in the five state-of-the-art competitive solutions.

KEYWORDS: UAV path planning, Sensor data collection, ACO, K-means, Wireless sensor network, Clustering, Power consumption.

1. Introduction

In recent years, the utilization of Unmanned Aerial Vehicles (UAVs) has gained considerable recognition from both the research community and industry, primarily because of their adaptability, efficacy, and affordability in various domains like agriculture, environmental monitoring, disaster relief, search and rescue, surveillance, and military operations [59]. In addition to their versatility, UAVs can offer real-time data collection (DC) capabilities, making them a valuable tool for various applications. Their popularity and extensive usage can be attributed to their affordability, mobility, dependable network access, and ability to establish line-of-sight links with ground Sensor Nodes (SNs) [58]. The integration of UAVs and Wireless Sensor Networks (WSNs) in large-scale areas has been extensively investigated by researchers. In this setup, UAVs act as mobile sinks that collect data directly from SNs or indirectly via Cluster Heads (CHs). The UAVs then transmit the gathered data to the Base Station (BS) for further analysis [11].

A sensor node periodically measures physical phenomena and is often powered by a small battery that can be challenging to replace or recharge when depleted. Therefore, preserving battery energy is of utmost importance and represents a primary research interest [24]. It stands to reason then that the energy consumption of the SN be kept to a minimum during DC, a concern fully addressed by the present article. On the other hand, the data measured by the SN has to be collected and transmitted to a «sink», where it can by analyzed, processed and/or stored. Many DC solutions have been proposed and can be generally divided into three basic types: WSNs, mobile sink (MS), and UAV.

The first DC basic solution uses a WSN that links the SNs wirelessly [19]. The SNs work together in a store-and-forward manner to transmit the measured data to the sink. Each SN acts as a data gatherer and a router. If all SNs are within range of at least one other, the network is fully connected, and data reaches the sink. However, if not, a relay node must be set up for complete connectivity. The advantage of a WSN solution lies in its ability to instantaneously transfer data to the sink. This low latency is highly valuable for real-time applications. However, if real time DC is not a critical factor, utilizing a WSN may not be advisable due to its many drawbacks [15].

First, load imbalance between SNs, where SNs close to the sink forward more data and lose energy faster. Clustering with a CH serving as a local sink can remedy this weakness. Second, WSN can come to a halt if a crucial SN becomes dead (e.g., due to battery depletion). Third, collisions can occur between neighbor SNs if transmissions are not coordinated, which can be avoided through coordination or retransmission. Fourth, WSN communications protocol requires non-trivial computations and communications, exhausting SN battery. Fifth, WSN requires full connectivity, which may require adding redundant SNs or relay nodes that may not be practical or feasible. Sixth, SN communications can be hampered by obstacles on the ground, requiring relay nodes for increased transmission range. Finally, election of a CH is problematic and energy-consuming, and a CH represents a single point of failure for the WSN. CHs are likely to lose energy and fail quickly due to their intensive work.

The second DC solution involves a mobile station, like a laptop, that collects data from nearby sensor nodes [51]. This approach brings the sink to the data instead of the other way around. The MS can be controlled by a person, animal, or vehicle. This solution offloads computations and communications from the SNs to the MS, eliminating communication between SNs [51]. The advantages of this solution are as follows. 1) Single hop data transmission ensures equal load for all SNs. 2) No risk of a single point of failure since there are no CHs. 3) Death of any SN affects only that SN, not the entire DC system. 4) No fear of collisions as the MS can inform each SN of when to transmit. 5) No need for SNs to perform communication protocol computations as transmission is single hop to UAV. 6) No need to install relay nodes as SNs do not need to be fully connected. 7) MS can improve data latency and save energy by stopping at points where data from multiple SNs can be collected at once.

The many-to-one data collection mode requires SNs to be within transmission range. However, the MS solution faces two problems: long routes resulting in high energy consumption and latency. These issues stem from non-linear movement between collection locations and impractical ideal collection points. The solution to this problem is a UAV, allowing the MS to fly and solve the issue of excessive route length.

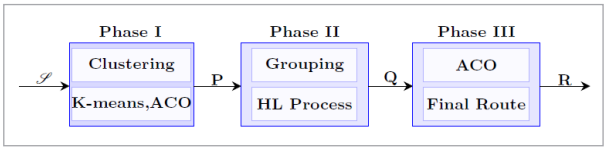
In this article, the UAV solution allows for data to be collected from SNs in a single hop from the air. The UAV flies high enough to avoid obstacles but low enough to be within transmission range. This has advantages over the MS solution, including line of sight (LoS) communications and extended transmission range, which can collect from multiple SNs at once [17]. The UAV's straight-line movement can drastically shorten the collection route, saving energy and reducing latency. Additionally, there are no unreachable points in the air, making it possible to collect data from any SN or DC point, particularly in the many-to-one collection mode.

Apart from the three basic solutions described above, there are also hybrid DC solutions combine WSNs with an MS or UAV for better performance. For example, in [3], an MS-assisted WSN is proposed, where the SNs are grouped into clusters, with each cluster having a CH, and the MS collects data solely from the CHs. Similarly, [6] employs a similar setup and focuses on developing an optimal route for the MS. Initially, Particle Swarm Optimization (PSO) is utilized to establish optimal cluster formation. Following cluster formation, the optimal number of data collection points are chosen, and a data-gathering route is planned for the MS.

The trouble with hybrid solutions is that they inherit the disadvantages of both of their components: the WSN and the complement, be it MS or UAV. So, they should in general be avoided unless necessary, as when the CHs are not in the transmission range of one another. To upload data from many SNs to the UAV fixed at some hovering location (HL), some studies consider the orthogonal frequency division multiple access (OFDMA) technique as a communications scheme, which allows the UAV to collect data from multiple SNs within its communication range simultaneously. For example, in [14], using the OFDMA the authors first formulate a DC maximization problem via deploying an energy constrained UAV and show the NP-hardness of the problem. Moreover, in [12], the authors also consider OFDMA subject to the energy capacity of the UAV. In the present article we propose a simple TDM scheme to achieve the same result. The scheme can be implemented in software, without requiring any hardware installation or modification.

In the present work, we propose a UAV DC solution for a wide sensory area (e.g. an agricultural farm), where $N \gg 1$ SNs are installed. The objective of the work is two-fold: a minimum UAV flight time and minimum SN computations and communications. The former ensures very low latency for the data, and very low energy consumption by the SNs, and is attained by a minimum UAV DC route. The latter ensures very low energy consumption by each SN and is attained by the UAV's collecting data directly from the SN in a single-hop. To this end, the solution goes through three phases, as shown in Figure 1.

Figure 1
The three phases of the proposed solution



In Phase I, we obtain an initial shortest route $\mathbf{P}$ traversing the N SNs. To this end, we first partition the N SNs into clusters, based on their physical positions, then use ant colony optimization (ACO) to determine the shortest route within each cluster. Clustering is employed to increase the accuracy and accelerate the convergence of ACO in its effort to find a shortest route traversing all N SNs. In Phase II, we partition the N SNs along the initial route $\mathbf{P}$ into groups each of which can fit in the footprint of the UAV while flying on top. For each group we identify the centroid, which is used as a HL, where the UAV will stop and collect the data of the group lying in the footprint. The sequence made up of the HLs forms the intermediate UAV route $\mathbf{Q}$. To improve the hovering route even further, its nodes are input to a suitable metaheurisitic such as ACO or Genetic Algorithm (GA) to permute them in the hope that a final shorter hovering route $\mathbf{R}$ is obtained.

The proposed solution has been implemented and tested for its validity and correctness. It has been found to largely produce routes that get progressively shorter, namely $|\mathbf{P}| > |\mathbf{Q}| > |\mathbf{R}|$, where $|\mathbf{X}|$ is the length, e.g., in meters, of route. When the results of the proposed solution, namely the length $|\mathbf{R}|$ of the final UAV route, are compared with those of state-of-the-art solutions, the proposed solution comes out superior. In addition, the proposed solution has many additional advantages including:

1 All the computations (initial route, intermediate route, final route, and collection lists) are done on

premise, rather than on the UAV or SNs, saving the energy of the latter and also ensuring low run time thanks to the more powerful computational resources usually available on site.

2 Only one transmission by the SN per data packet upload is needed, saving SN energy.

3 Only one reception by the UAV per data packet, saving UAV energy.

4 No risk of transmission collisions between SNs, thanks to the proposed TDM upload scheme.

5 No communications protocols or overhead is needed, as SNs communicate directly to the UAV, saving the energy of the SNs.

6 All SN data is guaranteed to be collected, and no single SN will be left out in a DC trip.

7 DC is completely distributed, meaning no single point of failure, e.g. cluster head, anchor node.

8 Equal energy load all SNs, as each is requested to transmit only its own data.

9 Line of sight communications are guaranteed; hence the longest and most reliable transmission range is guaranteed, which in turn leads to a large

The rest of the article is organized as follows. A review is presented in Section 2 of recent published work related to the subject. In Section 3 the system model and proposed solution are provided in full detail, including three algorithms to generate the three mentioned routes, are presented, and discussed. In Section 4, the experimental work is presented, where numerical results for some example configurations are obtained, analysed and discussed. In the final section, we give our concluding remarks.

2. Related Work

In a UAV DC solution, such as the one proposed in the present work, the energy consumption of the SNs is no longer an issue. Simply, each SN will transmit its data directly to the UAV in one hop, obviating the need for SN to SN communications and ancillary computations. As such, as most of the communications and computations chores are offloaded to the UAV, which does not have the energy scarcity problem the SNs have [48]. For one, the UAV usually has a large battery, enough to supply all the energy needed for DC. For another, this battery can be easily replaced or recharged once the UAV returns back to its base from a DC mission. Therefore, what really still poses a challenge in a UAV DC solution is minimizing the energy consumed by the UAV to keep it flying, and the only way to mitigate this challenge is to decrease the UAV flight route, which is the objective of the present article.

Given its profound importance, planning a short route for a UAV intended to collect data from SNs has attracted much research work. In [16], a strategy for an optimized UAV route is introduced for a disaster field. The field is divided into multiple cells and the authors formulate and solve two complementary subproblems: one identifying a minimal number of HLs at which the UAV hovers to collect data from all the SNs in the cell, and one constructing the UAV route that traverses those locations.

In [54], a UAV is dispatched to collect a given amount of data from some SNs. The authors consider two UAV routes, namely circular flight, and straight flight. In each case, the authors first derive the energy consumption expressions of the UAV and SN, and then find the optimal SN transmit power and UAV route that achieve a Pareto optimal trade-off. In [7], the authors formulate a non-convex optimization problem to maximize the minimum residual energy of the SNs after data transmission. To solve this problem, the authors first derive a feasible solution for the shortest UAV route, where a Voronoi diagram is modified to find a set of UAV HLs. Then with the initial shortest UAV route, a UAV route is proposed by adjusting each UAV HL sequentially based on SN energy status. In [30], the authors accept partial DC, unlike the case in the present article, where full DC is targeted. They investigate two problems: (1) optimizing the route of the UAV to minimize its travel time and DC time while guaranteeing the collection of a certain amount of data; and (2) finding the optimal route of UAV to maximize the minimum ratio of the collected data to the data stored in the SNs.

In [46], UAV route planning in the context of target detection is investigated based on integral geometric theory. The authors theoretically derived the target detection probability for both static and mobile target scenarios. In [8] deep deterministic gradient decent is used to autonomously decide the best UAV route to adopt in an obstacle-constrained environment, while Q-learning is used to determine the order of nodes to visit such that the DC time is minimized. In [13], HLs

are identified such that the UAV is able to collect data from as many SNs as possible from the same locations. The authors jointly consider the HL of the UAV and the utility maximization of DC, by first formulating a DC utility maximization problem (UMP) and show that it is an NP-hard problem. They devise an algorithm for positioning (potential) UAV HLs, which improves the DC utility.

In [28], the authors first divide the region into multiple cells, then design the flight routes for single UAV and multiple UAVs to cover all the cells. The per-node capacity of SN is derived as a function of the number of cells, the height of UAV, the number of SNs and the energy capacity of UAV. In [33], an optimal UAV DC route scheme based on matrix completion is proposed. Simulated annealing is used to plan the route of UAV based on the selected sampling points. In [50], the authors consider the problem of data loss, as some SNs may run out of storage space as a result of failing to upload their data to the UAV for an extended period of time. To mitigate this problem, a joint user scheduling and route planning DC strategy is formulated as a non-convex optimization problem which is then solved sequentially.

In [56], a farm made up of greenhouses of different sizes, with each having a number of SNs, is considered. To collect data from all SNs, the UAV flies along an optimal route generated by a genetic algorithm. The UAV height is controlled so that all the SNs of a greenhouse, regardless of size, can transmit their data reliability. In [40], the authors jointly optimize the route of a UAV and the radio resource allocation to maximize the number of sampled SNs, under the constraint that each SN having a data upload deadline. The formulate optimization problem is shown to be mixed integer non-convex and generally NP-hard, and solve it by a branch, reduce and bound algorithm. Route planning is also investigated for hybrid MS/WSN solutions, such as in [20], where a heuristic tour- planning algorithm is developed to find the shortest touring route for the MS to visits all SNs and collect data single hop from each. In [27], Q-learning is used to find the shortest route connecting the CHs. The study in [10] takes into account the amount of data in each SN. It constructs the shortest MS touring route and also dynamically adjusts the transmission rate for each SN based on the amount of its data. In [18], an ACO-based method is employed to select the best set of DC points and uses them to form the touring route for the MS. SNs sense the location of the MS and the duration it remains in transmission range to transfer all their data packets.

Route planning is also investigated for hybrid UAV/WSN solutions, such as in [24], where the UAV flays over each CH in an optimal route obtained by ACO. In [30], a pre-configured UAV route is planned using ACO to fly through each CH to gather data. Data compression takes place at the intermediate SNs to decrease upload time, but it of course adds an extra computation cost. In [41], the cross-edges approach and Kruskal's algorithm are used to generate the UAV route. The SNs in each cluster communicate using the ZigBee/IEEE 802.15.4 standards in the 2.4 GHz frequency band, which increases the danger of transmission collisions between them.

In [34], a DC strategy in UAV-aided WSNs for hilly regions is introduced using a UAV as a data mule. The UAV broadcasts beacon messages to the SNs to locate their CH, but this of course adds communications cost on the SNs. In [59], a deep reinforcement learning (DRL) method for solving the UAV route planning problem is employed. However, in contrast to LoS, the multi-route approach utilized in the ground network consumes more power and does not guarantee the longest and most reliable transmission range. In [11], a study is presented to minimize the WSN's energy consumption while satisfying the UAV route length requirement in a data collecting scheme for a heterogeneous WSN. In [53], the authors propose two working modes: single- and multiple-UAV scenarios for small-scale and large-scale DC systems, respectively.

In [6], the authors model the UAV motion as a probabilistic travelling salesman problem (PTSP), where the number of SNs to be served each time is a random variable. To optimize the UAV route, the authors propose an exact Branch and Bound algorithm that provides an optimal solution through each set of SNs which occur with certain probabilities. In [57], route planning for UAV is based on spiral decomposition, focusing on the rapid route planning for large-scale SNs evenly distributed in the circular area. In [52], smooth route construction for multiple UAVs in WSNs is proposed. The authors first develop a TSP based route construction algorithm, then extend it with route adjustments based on the required contact time at each SN. In [49], the authors propose an algo-

rithm based on grid division, to increase the efficiency of UAV route planning, while guaranteeing the length of the route to be short. In [35] a collaborative UAV-WSN network for monitoring large areas using a heterogeneous multi-agent scheme.

A mixed-integer based optimization procedure is employed into the associated constrained optimization problem. In [38], a hierarchical structure based on the collaboration between UAVs and federated WSNs for crop monitoring in precision agriculture is presented. In [60], the authors consider that the SNs which may or may not be within the transmission range of each other. Accordingly, they find a route for the UAV that includes HLs and determine the duration at each hovering such that the cumulative volume of data collected is maximized, subject to the energy capacity on the UAV. Two DC maximization problems, for full and partial collection, are formulated and solved by heuristics.

The research presented in [25] offers a comprehensive review of contemporary methodologies employed to improve the energy efficiency (EE) of UAVs. These methodologies encompass diverse aspects such as trajectory planning and deployment, resource allocation and management, design of energy-conserving communication protocols, as well as energy harvesting (EH) and transfer. Additionally, this paper extensively investigates pertinent research literature, thereby introducing several promising research directions for future exploration. In [42], authors propose a cluster-based routing approach to enhance UAV coverage with visual sensors. The model consists of four modules: online path planning, clustering-based topology construction, reinforcement learning-based cluster management, and data routing. The dynamic path planning algorithm determines UAV waypoints, while topology construction includes initialization, cluster head election, and formation. SARSA determines the optimal re-clustering policy for cluster management. Inter-cluster forwarders and selective route request flooding improve packet delivery and reduce delay. In [2], authors optimize UAV-assisted cluster-based WSNs in a 3D environment to enhance lifespan. Varying UAV altitude significantly affects lifetime and throughput. The proposed optimization redirects UAVs to efficient altitudes, outperforming centered placement at lower altitudes in terms of system lifetime.

In [39], authors presented a data collection and scheduling framework for smart farms. It involves two phases: data collection and scheduling. IoT sensors form clusters based on RSSI, allowing the UAV to collect data optimally. The UAV transfers data to the nearest base station. The base station selects an efficient fog node for workload processing. The framework was implemented in OMNeT++ and compared to existing approaches in terms of energy and network delay. In [5], the authors proposed an energy-efficient method for data gathering in deadline-based WSNs using multiple UAVs). The method optimizes UAV position, trajectory, travel time, and the number of UAVs required for efficient data collection. Simulation results show that the method achieves optimal performance in terms of energy consumption, travel time, and UAV utilization. In [23], the authors presented a cutting-edge method for energy-efficient clustering and cluster head selection in NG-WSNs. Their approach seamlessly integrates various components, including the midpoint technique, uniform sensor distribution, multihop communication, and the inclusion UAV within the network architecture. By leveraging these elements, along with the implementation of a simulated annealing algorithm for UAV trajectory optimization, the proposed approach exhibits remarkable superiority in terms of both energy efficiency and network lifetime when compared to existing techniques.

In [4], a single UAV was employed to optimize trajectory, reduce energy consumption of ground sensors in wireless networks, and maintain QoS and power constraints. The study incorporates two channel models for 4G, 5G, and B5G systems and investigates three trajectory optimization strategies. Namely exhaustive search, particle swarm optimization, and fixed placement, to locate the optimal trajectory of the UAV.

In [23], the authors introduced a framework to optimize UAV trajectory planning for energy-efficient data collection from IoT sensor nodes. It employs a data similarity-based node selection approach in three phases: data similarity determination using SDTW, redundant node removal with HGACA, and UAV trajectory planning through an ILP model. Simulation results show improved efficiency in execution time and power consumption while preserving data integrity, marking a substantial advancement in UAV data collection from IoT nodes.

In [44], the authors presented a comprehensive LiDAR dataset collected from vineyards in northern Spain using a DJI M300 UAV with a DJI Zenmuse L1 LiDAR sensor. The dataset includes high-density 3D LiDAR point clouds with embedded RGB information, serving various purposes such as optimizing vineyard management, aiding agricultural robotics development, and providing a "ground truth" dataset for validating satellite-derived models like digital elevation models (DEMs). It addresses the need for public UAV LiDAR datasets in Precision Agriculture, making it a valuable resource for the field.

In [36], the authors investigate surveillance as a security solution, focusing on heuristic neural analysis through artificial intelligence and deep learning. Automatic analysis of surveillance video content involves object/people tracking, detecting suspicious behavior, and sound analysis. In [58], a machine learning-based image analysis system combines optical flow and convolutional neural networks to recognize and track objects, especially detecting sudden movements and unfamiliar factors. In [26], the authors introduced an automated guided vehicle (AGV) technology which uses a combination of artificial intelligence and deep learning techniques, allows for the detection and identification of pallets for the purpose of automatically guiding the guided vehicle.

In [21], one example of the use of automated guided vehicles (AGV) in port environments is presented to handle and transport goods in a collision-free and safe path to avoid obstacles and arrive accurately at the shipping station using the star algorithm.

3. System Model

The objective of the present work is to collect the data of $N > 1$ SNs deployed in a sensor field using a UAV, in the shortest time possible to ensure both low latency for the data and low energy consumption by the UAV. Assuming a constant flying speed, the shortest time corresponds to the shortest flight route, which will be achieved in three phases, each having an algorithm, as described below. DC time, being mainly radio propagation time, will be ignored with respect to UAV flight time. Another objective, that is equally important, is to collect the data with a minimal amount of energy on the part of the SN, by minimizing the computational and communications chores of the SN. The N SNs are arbitrarily given the IDs $s_1, s_2, s_3, ..., s_N$.

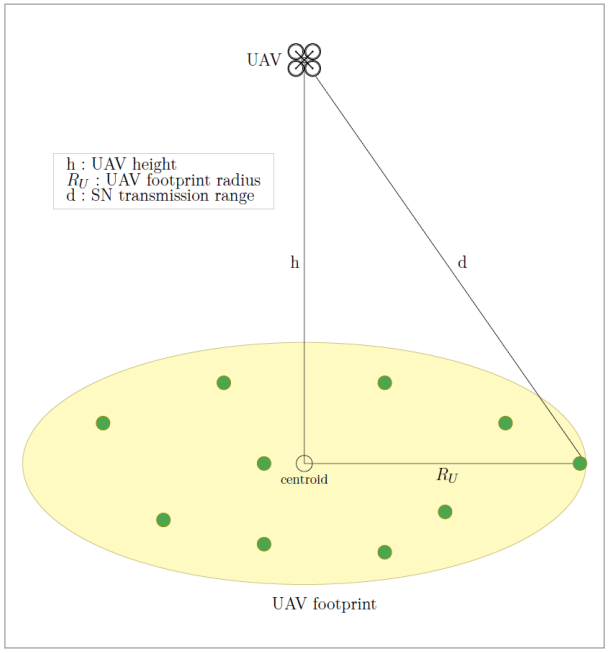
Let $\mathcal{S} = \{s_1, s_2, s_3, ..., s_N\}$ be the set of all SNs. The location of each SN s_i is determined by the ordered pair (x_i, y_i), where x_i and y_i are the Cartesian coordinates measured from some reference point, e.g. the lower left corner of the sensor field. It is assumed that there is a lookup table T having the position (x_i, y_i) of each SN s_i, and that this table is accessible to any component of the solution, e.g. the UAV and the algorithms. The set $\mathcal{S}$ of SNs will be partitioned into clusters to find an initial (SN) route, then will be partitioned again, this time along the initial route, to find an intermediate (hovering) route. The latter will be optimized to produce the final (hovering) route. By collecting data from many SNs in each UAV stop, called many to one DC, there will be a fewer HLs, hence a shorter UAV flight distance, realizing the objective of the solution.

3.1. Definitions

The definitions below are employed in the sequel.

- Hovering location: The point on the ground above which the UAV will hover to collect data from the SNs underneath.
- UAV footprint: The fixed size disk entered at the HL of the UAV, within which any SN can communicate with the UAV as shown in Figure 2. While the footprint size is fixed, the number of SNs it contains may differ from one footprint to another.
- Data packet: A fixed size data record transmitted by the SN to the UAV while above the hovering location. Its header contains the SN ID.
- Time slot: The amount of time needed to transmit a data packet.
- Collection window: The amount of time the UAV hovers over a footprint. It is equal to the number SNs in the footprint multiplied by the slot time.
- UAV packet: A packet transmitted by the UAV each time it is above a HL to make its presence known to the SNs in its footprint. It contains the IDs of those SNs in the footprint and the order in which they should to transmit their data so as to avoid collisions. It serves also as synchronization signal so that time slots in the collection window are properly recognized.

Figure 2
UAV footprint encompasses all SNs (green dots) that can transmit successfully to the UAV while hovering above the centroid (footprint center)



SN transmission range d: The maximum distance a wireless signal transmitted by the SN can reach when communicating line of sight (LoS). This range should be longer than in the case of non-LoS, e.g. in WSN or MS DC solutions, where obstacles may exist.

In addition, the following variables are used in the article:

N	Number of deployed SNs
m	Number of clusters
N_i	Number of SNs in cluster i
n	Number of groups (Number of hovering locations)
M_i	Number of SNs in group i

Clearly, the number of deployed SNs is

$$\sum_{i=1}^{m} N_i = \sum_{i=1}^{n} M_i = N.$$

3.2. Preliminaries

In what follows we explain a number of elements used in the proposed solution

3.2.1. Radius R_u of the UAV Footprint

Referring to Figure 2, the radius R_U of the UAV footprint is a function of its height above the HL and the transmission range of the SNs. Given the transmission range d of the SN, the UAV height h, then the UAV footprint radius R_U is given by

$$R_U = \sqrt{d^2 - h^2}. \quad (1)$$

It is desirable to have as wide a footprint as possible in order to enclose as large a number of SNs as possible and collect data from them in one UAV stop. This of course will decrease the number of HLs and consequently the UAV route and flying time. Since the transmission range d is constant for all the SNs, per assumption, the only way to widen the UAV footprint can be brought about by decreasing its height. However, to avoid ground obstacles and to ensure LoS communications with the SNs, there is a minimum height h that the UAV cannot fly below. Thus, we may define h to be the lowest height the UAV can fly at to avoid ground obstacles, which may hamper the flight or prevent LoS communications between the UAV and SNs on the edge of the footprint.

3.2.2. Shortest SN Traversal Route P and the ACO Algorithm

The ACO algorithm is used in Phase I of the proposed solution to find a shortest traversal route within each cluster and is also used in Phase III to optimize the UAV DC route. The decision to employ ACO is based on a careful consideration of the specific characteristics of our paper. In our paper, the total area of interest is subdivided into smaller regions using the k-means clustering technique. Within these smaller areas, ACO has consistently demonstrated superior performance in obtaining the shortest path when compared to some other algorithms like Genetic Algorithms (GA). To further substantiate our choice, we conducted experiments that provide empirical evidence supporting the effectiveness of ACO in this paper.

Capable of solving combinatorial optimization problems, ACO takes inspiration from the behaviour of real ant colonies [55]. When searching for food, ants put a substance called pheromone on the ground in an effort to make the shortest route to the food source known for other ants. Clearly, the more ants that use a particular trail, the more pheromone will

be on there, which in turn attracts more ants to use that trail. However, pheromone is not a permanent substance but can evaporate. Thus, if a trail has some pheromone but is ignored by ants for a while, whatever pheromone there will evaporate with time, turning off any ant that wanted to use the trail. Then, ants that search later for food trace the presence of pheromone and follow the trait that has a higher concentration of the substance. Once at the food source, the ants are able to get food and return back to their colony using the same trail, dropping more pheromone which makes the trail ever more popular as time goes by.

ACO algorithms act similarly, using a number of artificial ants—tiny computational agents, that work cooperatively and communicate through artificial pheromone trails. In the computer replica of ant behavior, the equivalent of time is iterations. In particular, and regarding our particular problem, a number v of artificial ants are used to search for the shortest path traversing a set of N nodes starting at a given node. Every ant constructs a solution to the problem by travelling on a constructed graph. In each iteration, every ant makes a move to one and only one neighbor node. During their tour, all ants leave an amount of pheromone on the edges they have cross, so as to help the ants that will use the edges next. When an ant is at a given node i, it selects the next node j based on the amount of pheromone τ_{ij} and the heuristic desirability η_{ij} of the trail (i,j) connecting the two nodes. Specifically, the probability for the ant to go from node i to node j is a function node of the two quantities τ_{ij} and η_{ij} raised to the powers α and β which are two parameters determining the relative influence of the pheromone and the heuristic information, respectively. At the end of each iteration, the amount of pheromone on each trail is updated based on the amount that has been dropped and the amount that has evaporated during the iteration.

3.2.3. Clustering SNs and the K-means Algorithm

The K-means algorithm is used in Phase I of the proposed solution to partition the set $\mathscr{S}$ of SNs into non-overlapping clusters. Based on unsupervised learning, it places in the same cluster all the points with similar features or characteristics, using the least squares concept as follows. First, it receives as input the number k of required clusters. Then it selects randomly k points representing an initial k centroids without replacement. It keeps iterating until there is no change to the centroids, i.e assignment of points to clusters is not changing. Now, it computes the sum of the squared distances between the points and all centroids, and assigns each point to the closest cluster (centroid). Finally, it computes the centroid of a cluster by taking the average of all points that belong to each cluster. The centroid of $k \succ 1$ points $(x_1, y_1), (x_2, y_2), ..., (x_k, y_k)$ deployed in the Euclidean plane is also a point $c = (x, y)$ in the same plane, with x and y given by

$$\bar{x} = \frac{x_1 + x_2 + ... + x_k}{k}, \tag{2a}$$

$$\bar{y} = \frac{y_1 + y_2 + ... + y_k}{k}. \tag{2b}$$

The total mean-square quantisation error (MSE) [1] is used as a fitness function of the K-means algorithm.

3.3. Solution Phases and Algorithms

Clearly, the start SN p_{1_1} of cluster 1 will also be the start SN p_1 of entire set $\mathscr{S}$ of deployed SNs. The proposed solution is comprised of three phases, as shown in Figure 1, each having a well developed algorithm and ending up with a route that gets (hopefully) shorter with the phases.

Algorithm 1: Initial (SN) route

Input: Set $\mathscr{S}$ of N SNs, Start SN p_{1_1}, Desired number m of clusters, SN position lookup table.

Output: Initial (SN) route $\mathbf{P} = (p_1, p_2, ..., p_N)$.

1 $\mathbf{P} := []$ //Initialize the initial route.
2 Apply the K-means algorithm to the set $\mathscr{S}$ to partition it into m clusters: $\mathscr{C}_1, \mathscr{C}_2, ..., \mathscr{C}_m$.
3 For each cluster $\mathscr{C}_i$ find its centroid, as per (2), placing all m centroids in some set φ.
4 Denote by c_1 the centroid of the cluster, where the start SN p_{1_1} exists, and by N_1 the cardinality of that cluster.
5 Delete the centroid c_1 from the set φ.
6 Apply the ACO algorithm to the cluster with centroid c_1 to obtain a shortest path sub-route $\mathbf{P}_1$ $P_1 = (p_{1_1}, p_{1_2}, ..., p_{1_{N_1}})$ traversing the cluster.
7 Append sub-route $\mathbf{P}_1$ to the initial route $\mathbf{P}$.

8 **for** $i = 1$ to $m-1$ **do**

9 Calculate the distance from SN $p_{i_{N_i}}$, the end SN of sub-route $\mathbf{P_i}$, to every centorid in the set φ.

10 Denote by c_{i+1} the centroid closest to $p_{i_{N_i}}$, and by N_{i+1} the cardinality of the cluster of that centroid.

11 Delete the centroid c_{i+1} from the set φ.

12 Calculate the distance from SN $p_{i_{N_i}}$, the end SN of sub-route $\mathbf{P_i}$, to every SN of the cluster with centroid c_{i+1}.

13 Denote by p_{i+1_1} the SN closest to $p_{i_{N_i}}$.

14 Apply the ACO algorithm to the cluster with centroid c_{i+1}, starting at SN p_{i+1_1}, to obtain a shortest path sub-route $\mathbf{P_{i+1}} = \left(p_{i+1_1}, p_{i+1_2}, ..., p_{i+1_{N_{i+1}}}\right)$ traversing the cluster.

15 Append sub-route $\mathbf{P_{i+1}}$ to the initial route $\mathbf{P}$.

16 **end**

Initial (SN) route $\mathbf{P}$: In phase I, an initial SN traversal route $\mathbf{P} = (p_1, p_2, ..., p_N)$ traversing all N SNs in a shortest route manner is constructed from the set $\mathscr{S}$ of all SNs, using both K-means clustering and ACO. Algorithm 1 takes $\mathscr{S}$ as input and gives $\mathbf{P}$ as output.

Intermediate (hovering) route $\mathbf{Q}$: In Phase II, an intermediate UAV hovering route $\mathbf{Q} = (q_1, q_2, ..., q_M)$ is constructed from the initial route $\mathbf{P}$. This is done by grouping the SNs along $\mathbf{P}$ into n groups (mutually disjoint subsets) each of which can fit in the footprint of the UAV as it hovers above in the air. Algorithm 2 takes $\mathbf{P}$ as input and gives $\mathbf{Q}$ as output. Effectively, this phase partition $\mathbf{P}$ into M collection lists $\mathbf{L_1}, \mathbf{L_2}, ..., \mathbf{L_M}$, each of which can fit in the footprint of the UAV. Clearly, $n \le N$ and $|\mathbf{Q}| \le |\mathbf{P}|$, with the equality holding only if each group contains one and only one SN.

Final (hovering) route $\mathbf{R}$: Finally, in Phase III a final hovering route $\mathbf{R}$ obtained by optimizing (rearranging the elements of) $\mathbf{Q}$ using ACO, if the number of SNs is large, or GA if that number is small. That is, $\mathbf{R}$ is produced by recording the elements, i.e. the HLs, of $\mathbf{Q}$. Algorithm 3 takes $\mathbf{Q}$ as input and gives $\mathbf{R}$ as output. Clearly, $|\mathbf{R}| \le |\mathbf{Q}|$, with the equality if the optimization algorithm, ACO or GA, receives an already optimal route.

3.3.1. Phase *I*: Initial (SN) Route P

In this phase, the ACO algorithm is applied to the set $\mathscr{S}$ of all deployed SNs in order to construct the shortest route through the SNs. However, doing so would produce two problems. First, the chances of ACO's finding the shortest path would be very slim especially if the size of $\mathscr{S}$ is large. Second, ACO would take a large amount of time to finish its work, regardless of the final result. To avoid these problems, we start by partitioning $\mathscr{S}$ into an arbitrary number $1 < m < N$ of clusters using the K-means algorithm, and finding the centroid $c = (\overline{x}, \overline{y})$ of each cluster. Each cluster has a number N_i SNs, with $\sum_{\min}^{\max} N_i = N$. Then, we apply ACO to the clusters separately to obtain a shortest route within each cluster, and eventually concatenate these subroutes to obtain a short route traversing all the SNs. Needless to say, this route may or may not be the shortest, but it can be made very close to the shortest by knowing where to start the ACO in each cluster. By much experimentation, we have reached a method that makes the traversal route through $\mathscr{S}$ very short, compared to what has been reported in other research papers. The method starts in the second cluster, as the start SN for the first cluster is user defined.

After ACO produces the shortest route $\mathbf{P_1} = \left(p_{1_1}, p_{1_2}, ..., p_{1_{N_1}}\right)$ for the initial cluster, we proceed to determine the distances between the last SN $p_{1_{N_1}}$ on the route $\mathbf{P_1}$ and the centroids of all the other clusters, where N_1 is cardinality of the initial cluster. The subsequent cluster to be traversed is determined by the shortest distance.

Identify the closest SN in the subsequent cluster to the last SN p_{1N_1} on the route $\mathbf{P_1}$. The closest SN would then be the start SN for the second subroute $\mathbf{P_2}$, to be produced by ACO, and the cluster where it resides would be the second cluster. As depicted in Algorithm 1, repeat the above three steps for all the remaining clusters, getting at the end m sub-routes $\mathbf{P_1}, \mathbf{P_2}, ..., \mathbf{P_m}$. Construct the initial UAV route $\mathbf{P}$ by concatenating the $\mathbf{P_i}$ in order, i.e.

$$\begin{aligned}\mathbf{P} &= (\mathbf{P_1}, \mathbf{P_2}, ..., \mathbf{P_m}) \\ &= (p_{1_1}, p_{1_2}, ..., p_{1_{N_1}}, p_{2_1}, ..., p_{2_2}, ..., p_{2_{N_2}}, ..., \\ &\quad p_{m_1}, p_{m_2}, ..., p_{m_{N_m}}) \\ &= (p_1, p_2, ..., p_N)\end{aligned}$$

We can see that $\mathbf{P}$ is basically a passive [9] permutation σ, say, of the set $\mathscr{S} = \{s_1, s_2, ..., s_N\}$. If, for sim-

plicity, we consider σ to be a permutation of the set $\{1,2,...,N\}$, rather than the set of SNs, then for each $i \in \{1,2,...,N\}$, the i th element of the route $\mathbf{P}$ is just element $s_{\sigma(i)}$ of the set $\mathscr{S}$, i.e. $p_i = s_{\sigma(i)}$.

Algorithm 2: Intermediate (hovering) route

Input: UAV initial route $\mathbf{P} = (p_1, p_2, ..., p_N)$ and UAV footprint radius R_U.

Output: Intermediate route $\mathbf{Q} = (q_1, q_2, ..., q_n)$, where q_j is the j th location group, and collection lists $\mathbf{L}_1, \mathbf{L}_2, ..., \mathbf{L}_n$, where $\mathbf{L}_j$ is the list of SNs that will be collected while UAV at HL q_j.

$i := 1$; //Index of SN along the initial route.
$n := 1$ //Group no.
$L_n := [\,]$ //Initialize nth collection list with first SN in group.
$\mathbf{Q} := [\,]$ //Initialize intermediate (hovering) route.
$\mathscr{C} := \varphi$ //Initialize footprint group test set.
do
  $\mathscr{C} := \mathscr{C} \cup \{p_i\}$ //Attempt adding a SN to the current group.
  Calculate centroid c of set $\mathscr{C}$, as per (2).
  Flag := 0.
  foreach $s \in \mathscr{C}$ **do**
    Calculate distance d_s from SN s to centroid c.
    if $d_s > R_S$ **then**
      Flag := 1 // SN is outside footprint, so take action as below.
    end
  end
  if Flag = 0 then
    Append p_i to L_n //SN is inside footprint, so add it to collection list.
    if $i < N$ then
      $c_n := c$ //Retain previous centroid.
    end
    else
      Append c to $\mathbf{Q}$ //SN is the last one, so add current centroid to route and stop.
    end
  end
  else
    if $i < N$ **then**
      Append c_n to $\mathbf{Q}$ //SN outside footprint, so write previous centroid as route node.
      $n := n + 1$ //Close current group and start a new one.
      $L_n := [p_i]$ //Initialize a new list with the SN that did not join previous list.
      $\mathscr{C} := \{p_i\}$ //Initialize a new group set with the SN that did not join previous list.
    end
    else
      Append c to $\mathbf{Q}$ //SN is the last one, so add current centroid to route and stop.
      Append p_i to $\mathbf{Q}$ //Add last SN to route, as it is the centroid of itself.
      $L_{n+1} := [p_i]$ //Place the last SN in a new collection list.
    end
  end
  $i := i + 1$//Increment i to nominate another SN, or exit while loop.
while $(i \le N)$;

3.3.2. Phase *II*: Intermediate (Hovering) Route Q

The Intermediate Route algorithm takes the initial route $\mathbf{P} = (p_1, p_2, ..., p_N)$, and the UAV footprint radius R_U, as input and outputs an intermediate hovering route $Q = (q_1, q_2, ..., q_n)$ and n collection lists $\mathbf{L}_1, \mathbf{L}_2, ..., \mathbf{L}_n$, where q_i is the i th UAV HL and L_i is the list of SNs, in order, whose data will be collected by the UAV while at that location. The intermediate route Algorithm 2 effectively partitions the SNs along $\mathbf{P}$ into $n \prec N$ groups each of which fits in the footprint of the $\mathbf{Q} = (q_1, q_2, ..., q_n)$ UAV if it hovers over the centroid of the group. The algorithm achieves its aim by building one group at a time, starting at the first SN P_1 of $\mathbf{P}$. An empty test set $\mathscr{C}$ is created for each group, to which SNs are added one at a time.

Each time a SN is added to $\mathscr{C}$ its centroid will be calculated. The algorithm then compares the distance between the added SN and the centroid of $\mathscr{S}$ with the UAV footprint radius R_U. If the distance is less than R_U, then the SN is added to the current group, and

the algorithm moves on to attempt to add another SN. Else, the SN is not added to the group, which will then be closed and considered complete, and the algorithm moves on to build a new group, adding to it that SN as the first member.

The M_i members of group i will be inserted, in the order they are added to the group, in the collection list $\mathbf{L_i}$ of that group. The centroid of group i, denoted by q_i, will be considered the UAV HL for the group.

This exercise is continued until all N SNs along the initial route $\mathbf{P}$ are accounted for, noting that $\sum_{i=1}^{n} M_i = N$. The algorithm outputs the intermediate route $\mathbf{Q} = (q_1, q_2, ..., q_n)$, which is the vector of centroids of the n formed groups, with each centroid q_i associated with a collection list $\mathbf{L_i}$ telling which SNs will transmit data and in what order while the UAV is hovering above q_i. It is as if the Intermediate route algorithm partitions the initial route $\mathbf{P}$ into n lists $\mathbf{L_1}, \mathbf{L_2}, ..., \mathbf{L_n}$, or $\mathbf{P} = (\mathbf{L_1}, \mathbf{L_2}, ..., \mathbf{L_n})$,

where

$$\mathrm{L}_1 = (p_1, p_2, ..., p_{M_1}),$$

$$\mathrm{L}_2 = (p_{M_1+1}, p_{M_1+2}, ..., p_{M_1+M_2}),$$

...

$$\mathrm{L}_n = (p_{M_1+M_2+...+M_{N-1}+1}, p_{M_1+M_2+...+M_{N-1}+2}, p_{M_1+M_2+...+M_{N-1}+2}, ..., p_{M_1+M_2+...+M_N}) = (p_{N-M_m+1}, p_{N-M_n+2}, ..., p_N).$$

As an example, assume $\mathrm{P} = (3,5,1,7,9,2,4,8,6)$. Assume also that the Intermediate (hovering) route $\mathbf{Q} = (q_1, q_2, ..., q_n)$ algorithm was able, based on the locations of the SNs and the footprint radius, to form three groups, meaning that $n = 3$, with $M_1 = 4, M_2 = 2, M_3 = 3$. Then, the algorithm will produce $\mathrm{L}_1 = (3,5,1,7)$, $\mathrm{L}_2 = (9,2)$, $\mathrm{L}_3 = (4,8,6)$. That is, the UAV will collect the data of the SNs $(3,5,1,7)$ in this order, while hovering atop their centroid, then moves to the centroid of $(9,2)$ to collect their data, and finally moves to the centroid of $(4,8,6)$ to collect their data, completing the exercise of data transmission by the SNs and collection by the UAV.

3.3.3. Final (Hovering) Route $\mathbf{R}$ Algorithm

The Final Route algorithm, given formally in 3, takes the set $\Pi = \{q_1, q_2, ..., q_n\}$ of the elements of the intermediate route $\mathbf{Q} = (q_1, q_2, ..., q_n)$ and applies ACO to it, with the aim to shorten the length of that route. Thus, the output of this algorithm is basically some permutation γ, say, of the set Π, noting that each element q_i preserves with it its associated collection list $\mathbf{L_i}$. That is, as the elements of Π are permuted by γ, the collection lists will be permuted by the same γ. The algorithm outputs a final hovering route $\mathrm{R} = (r_1, r_2, ..., r_n)$, where $\mathbf{R}$ is basically some passive permutation γ of the set $\Pi = \{q_1, q_2, ..., q_n\}$. If, for simplicity, we consider γ to be a permutation of the set $\{1,2,...,n\}$, rather than the set of HLs, then for each $i \in \{1,2,...,n\}$, the ith element of $\mathbf{R}$ is just element $q_{\gamma(i)}$ of the set Π, i.e. $r_i = q_{\gamma(i)}$.

3.3.4. Time Complexity

Complexity analysis of Algorithm 1. For N SNs, $m < N$ clusters and number of itreations I, the time complexity of the K -means clustering algorithm in step 2 is given as $\mathrm{O}(\mathrm{mNI})$ [43]. The time complexity of ACO algorithm in step 14 is given as $O(\sigma N_{i+1}^2 I_{\max})$, where N_{i+1} is the cardinality of cluster $\mathscr{C}_{i+1}$, σ is the number of ants and $I_{\max}$ is the maximum iteration [37]. For $m < N$ clusters, the time complexity for steps 8 to 16 is given as $\sum_{i=1}^{m-1} O(\sigma N_{i+1}^2 I_{\max})$. The overall time complexity $T_{complexity}$ of the algorithm is given as

$$T_{complexity} = O(mNI) + \sum_{i=1}^{m-1} O(\sigma N_{i+1}^2 I_{\max}).$$

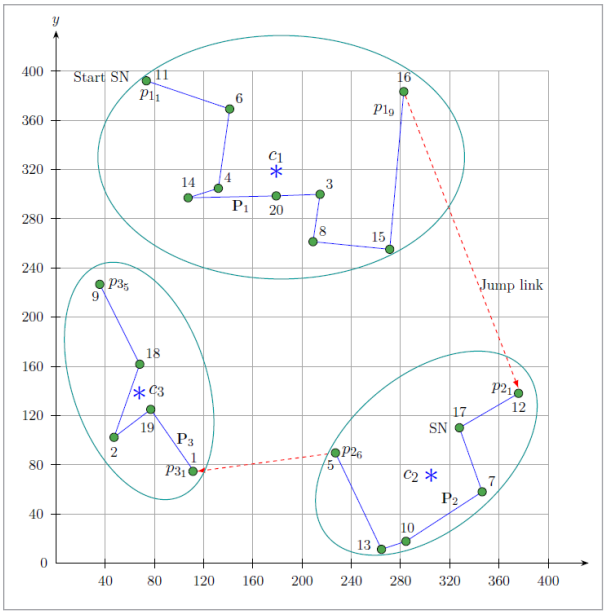
On the other hand the time complexity of Algorithm 2 can be analysed as nested loop (the outer loop and the inner loop), The outer loop (Step 6 to 39) in the algorithm iterates over each SN from the initial route $\mathbf{P}$. This loop has a time complexity of $O(N)$, the inner loop (Step 10 to 15) iterates over the elements in each SNs group. The overall time complexity of algorithm 2 can be approximated as $O(N^2)$. For algorithm 3, the time complexity is given as $O(n^2)$, where n is set of all SNs of the intermediate route $\mathbf{Q}$ and $n \ll N$.

Algorithm 3: Final (hovering) route

Input: Intermediate (hovering) route $\mathbf{Q} = (\mathrm{q}_1, \mathrm{q}_2, ..., \mathrm{q}_n)$, where q_j is the j th hovering location, and the corresponding n reading lists $\mathbf{L_1}, \mathbf{L_2}, ..., \mathbf{L_n}$.
Output: UAV final route $\mathbf{R} = (r_1, r_2, ..., r_n)$, where r_j is the j th hovering location, and the n collection lists $\mathbf{L_1'}, \mathbf{L_2'}, ..., \mathbf{L_n'}$.

1 $\Pi = \{q_1, q_2, ..., q_n\}$ //Set of all elements of the intermediate route $\mathbf{Q}$ into a set.
2 Apply the ACO algorithm to Π to find the shortest path $\mathbf{R}$ that traverses the points of Π starting at location q_1.
3 Relabel each reading list $\mathbf{L_i}$ to match the relabelling of each q_i to the corresponding r_j, $i, j = 1,2,...,n$.

Figure 3
Sensory field of area 400 × 400 m² with 20 SNs randomly deployed



3.4. Illustrative Example

We give now an illustrative example to show how the framework works till it finds the final (hovering) route $\mathbf{R}$.

In this example, the set $\mathscr{S}=\{1,2,3,...,20\}$ of 20 SNs are randomly deployed in an area of 400×400 m^2 depicted visually in Figure 3, with the lookup table $\mathbf{T}$ as shown in Table 1.

3.4.1. Generating the Initial Route P

As per steps 1-2 in Algorithm 1, the K-means clustering algorithm is told to divide the 20 SNs into $m=3$ clusters. The three clusters are $\{11,6,8,14,15,16,4,3,20\}$, $\{5,12,7,13,17,10\}$, $\{1,2,19,18,9\}$ as shown in Figure 3. The centroids of these three clusters when calculated are found to be $c_1=(178.88,317.96)$, $c_2=(305,70.75)$ and $c_3=(67.78,138.03)$, respectively.

As per the for loop in Algorithm 1, ACO will start working in the cluster containing this start SN, no. 11, i.e. the cluster with centroid $\mathrm{c}_1=\left(178.88,317.96\right)$. Invoking the ACO algorithm for the SNs of cluster 1, with SN 11 as a start SN, step 6 in the Algorithm 1 yields the sub-route

$$\mathrm{P}_1=\left(11,6,4,14,20,3,8,15,16\right), \tag{3}$$

Table 1
The lookup table which contains the SNs information

ID	Position	ID	Position
1	(111.27, 74.59)	11	(73.4,392.18)
2	(47.12, 102.24)	12	(375.89,138.02)
3	(214.58, 299.89)	13	(264.49,11.18)
4	(131.9, 304.77)	14	(107.34,297.04)
5	(227.09, 89.6)	15	(271.19,255.15)
6	(141.13, 369.26)	16	(282.61, 383.4)
7	(346.31, 57.96)	17	(327.82,110.02)
8	(208.85, 261.35)	18	(68.04,161.7)
9	(35.49, 226.68)	19	(76.96,124.96)
10	(284.4, 17.69)	20	(178.94, 298.58)

whose length, using the lookup table $\mathbf{T}$, is 500.05 m. Calculate the distances from SN 16 to centroid c_2 and c_3 which gives, 313.45 m and 326.13, respectively. The centroid c_2 of cluster 2 is the closest centroid to SN 16. To traverse cluster 2, we find in it the SN closest to the end SN of $\mathbf{P}_1$, namely 16, by calculating the distances between SN 16 and all the SNs of cluster 2. This calculation is shown in Table 2, from which we find that the subroute $\mathbf{P}_2$ of cluster 2 should start with SN 12 which has the smallest distance 262.51 m with SN 16. When the ACO algorithm is given $\mathscr{S}(e_2)=\{12,17,7,10,13,5\}$ and SN 12 as start SN as input, it outputs the subroute

$$\mathrm{P}_2=\left(12,17,7,10,13,5\right), \tag{4}$$

whose length, using the lookup table $\mathbf{T}$, is 292.56 m.

In a similar manner, to traverse cluster 3, we find in it the SN closest to the end SN of $\mathbf{P}_2$, namely 5, by calculating the distances between SN 5 and all the SNs of cluster 3 as shown in Table 3. This calculation shows that the subroute $\mathbf{P}_2$ should start with SN 1 which has the smallest distance $\left(116.79\text{ m}\right)$ with SN 5. When the ACO algorithm is given $\mathscr{S}(e_3)=\{1,2,19,18,9\}$ and SN 1 as start SN as input, it outputs the subroute

$$\mathrm{P}_3=\left(1,19,2,18,9\right), \tag{5}$$

whose length, using the lookup table $\mathbf{T}$, is 234.16 m. Referring to Figure 3, using step 12 in algorithm 1, the

initial route $\mathbf{P}$ is given by concatenating the three subroutes (3)-(5) getting

$$\begin{aligned}\mathbf{P} &= \left(\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3\right). \\ &= (11,6,4,14,20,3,8,15,16,12,17,7, \\ &\quad 10,13,5,1,19,2,18,9),\end{aligned} \tag{6}$$

whose length, by summing the lengths of the three subroutes and connection line lengths, is 1406.07 ≈1.4 km.

Table 2
Distance between the SN 16 to other SNs in cluster 2

ID	Distance(m)
12	**262.51**
17	277.09
7	331.62
10	365.71
13	372.66
5	299.0

Table 3
Distance between the SN 5 to other SNs in cluster 3

ID	Distance(m)
1	**116.79**
19	154.24
2	180.41
18	174.63
9	235.59

3.4.2. Generating the (Hovering) Intermediate Route Q

In this example, we set the UAV height $h = 80$ m and the UAV to the edge SN distance $d = 88$ m. Using Figure 2, the UAV radius R_U is calculated to 36.66 m. Applying steps 1 through 5 in Algorithm 2, $\mathbf{L}_1 := [\,]$, $\mathbf{Q} := [\,]$, and $\mathscr{C} := \varphi$. Then follow steps 7 and 8, the set $\mathscr{C}$ is updated to $\mathscr{C} := \{11\}$ with centroid $c_1 = (73.4, 392.18)$. Using step 11, the distance d_s from the SN 11 to the c is calculated to 0 m. Since d_s satisfies the condition in steps 16 in Algorithm 2, then $\mathbf{L}_1$ is updated to $\mathbf{L}_1 := [11]$.

Carrying out steps 6 to 39 yields, $\mathbf{L}_1 := [11,6]$. and $\mathscr{C} := \{11,6,4\}$ with centroid $c_1 = (107.3, 380.7)$. Carrying out steps 6 to 39 again shows that SN 4 along the initial route does not satisfy the condition at step 16. As a result, the collection list $\mathbf{L}_1$ is closed with $\mathbf{Q} := [(107.3, 380.7)]$. After executing steps from 6 to 39 in Algorithm 2 N times, the intermediate route is completely generated by adding all the resulting centroids to $\mathbf{Q}$, getting the elements of $\mathbf{Q}$ as shown in Table 4.

Table 4
The generated groups and their centroids

Hovering location	Group
$q_1 = (107.3, 380.7)$	(11, 6)
$q_2 = (119.6, 300.9)$	(4, 14)
$q_3 = (200.8, 286.6)$	(20, 3, 8)
$q_4 = (271.2, 255.2)$	(15)
$q_5 = (282.6, 383.4)$	(16)
$q_6 = (351.9, 124.0)$	(12, 17)
$q_7 = (346.3, 58.0)$	(7)
$q_8 = (274.4, 14.4)$	(10, 13)
$q_9 = (227.1, 89.6)$	(5)
$q_{10} = (94.1, 99.8)$	(1, 19)
$q_{11} = (57.6, 132.0)$	(2, 18)
$q_{12} = (35.5, 226.7)$	(9)

Figure 4, the composition of the set $\mathbf{Q}$ is visually depicted, wherein each element of $\mathbf{Q}$ corresponds to the centroid of a distinct set of SNs enclosed within individual circles. These centroids, originating from q_1, collectively delineate the starting points of the UAV HLs. The arrangement of these centroids gives rise to the intermediate route $\mathbf{Q}$. In essence, $\mathbf{Q}$ encapsulates the representation of UAV HLs initiated from q_1 and encompasses the spatial distribution of SNs within each circle.

The route length of the route $\mathbf{Q}$ can be easily calculated using Table 4 as 1155.95»1.2 km.

Figure 4
The intermediate route **Q** consists of 12 hovering locations of 12 groups

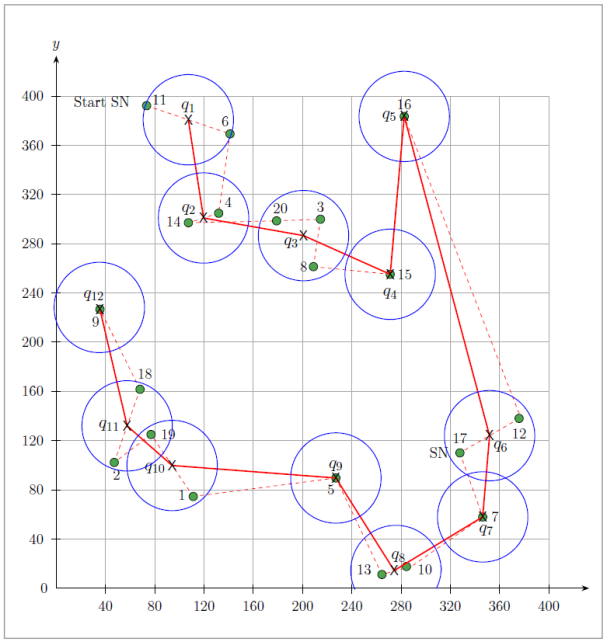
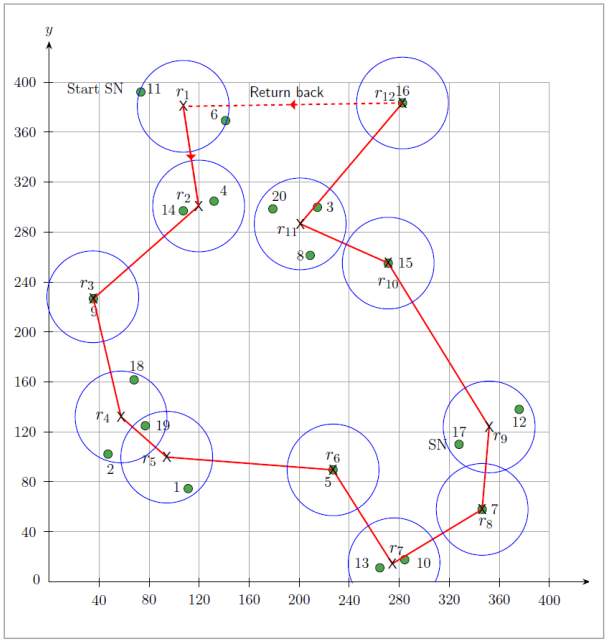


Figure 5
The final UAV route **R** for gathering data from 20 SNs by hovering 12 locations



3.4.3. Generating the Final Route R

For further UAV trajectory improvement, Algorithm 3 is utilised to generate the final route **R** based on the intermediate route **Q** formed in Figure 4.

This can be conducted using steps 1-3 in the Algorithm 3, where, the ACO algorithm is applied to obtain a new permutation of the HLs within the route **Q** to get the final route $\mathrm{R}=\left(r_1, r_2, r_3, \ldots, r_{12}, r_1\right)$ as shown in Figure 5.

Based on this figure and Table 5, the route length of the final route **R** is calculated using Table 5 as $1069 \approx 1.1$ km.

4. Experimental Work

To validate the proposed solution, it has been implemented by the authors in Python 3.9 and the code was used to perform several simulation experiments. A ready-made simulation package was avoided in order to ensure full control over every operational detail. The experiments were conducted on a PC with an Intel i7 processor @2.4 GHz and having 16 GB of main memory. Table 6 summarizes the operation-

Table 5
The final hovering locations and their associated groups

Hovering location	Group
$r_1=(107.3,380.7)$	(11, 6)
$r_2=(119.6,300.9)$	(4, 14)
$r_3=(35.5,226.7)$	(9)
$r_4=(57.6,132.0)$	(2, 18)
$r_5=(94.1,99.8)$	(1, 19)
$r_6=(227.1,89.6)$	(5)
$r_7=(274.4,14.4)$	(10, 13)
$r_8=(346.3,58.0)$	(7)
$r_9=(351.9,124.0)$	(12, 17)
$r_{10}=(271.2,255.2)$	(15)
$r_{11}=(200.8,286.6)$	(20, 3, 8)
$r_{12}=(182.6,383.4)$	(16)

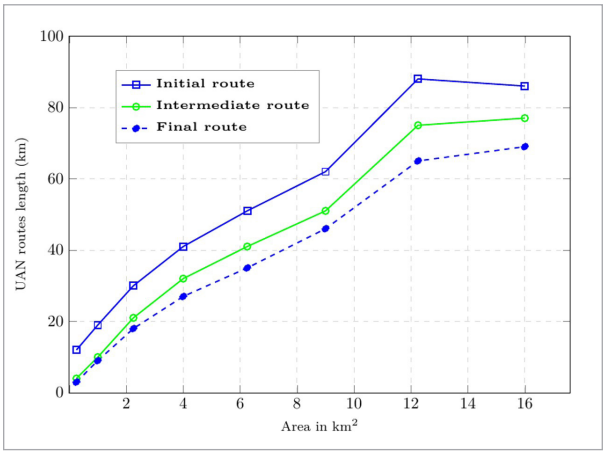
al simulation parameters. At first, we carried out an experiment to make sure that the three phases of the solution produce progressively shorter routes as intended, monitoring at the same time the operational parameters under which the solution performs best. In this experiment, we calculated the lengths of the initial route $\mathbf{P}$, intermediate route $\mathbf{Q}$ and final route $\mathbf{R}$ under different operational scenarios, with the results shown in Figures 6-7. The aim of the first experiment is to investigate whether the three phases of the framework mange to produce a progressively shorter route, regardless of the area of the sensory field. Figure 7 indicates that the answer is: yes. The figure shows the lengths of the three routes, initial, intermediate, and final, versus the area of the sensory field, when 500 SNs are randomly deployed in the field.

Table 6
Simulation parameters

Parameters	value
N	200, 500, 1000, 1500 SNs
m	1,2, . . . ,12 clusters
Monitoring Area (km^2)	4, . . . ,16
SN transmission range	50 m
AUV's height(m)	40, 60, 100 m
UAV footprint, $\boldsymbol{R}_u$	Calculated
Number of ants	20
Number of iterations	150, 250
The pheromone evaporation rate, ρ	0.05
The relative influence of the pheromone trail, α	1
Heuristic information, β	1

At the outset, we can see that the route length is a function of both the deployment area and the number of SNs, which means that we have two elements to investigate. First, we notice that the length of each of the three routes increases as the sensing area size increases. This is natural, for when the SNs are deployed all over a larger area, the route that traverses them will necessarily be longer.

Figure 6
The UAV routes length (km) in small and large scale area with N= 500 SNs

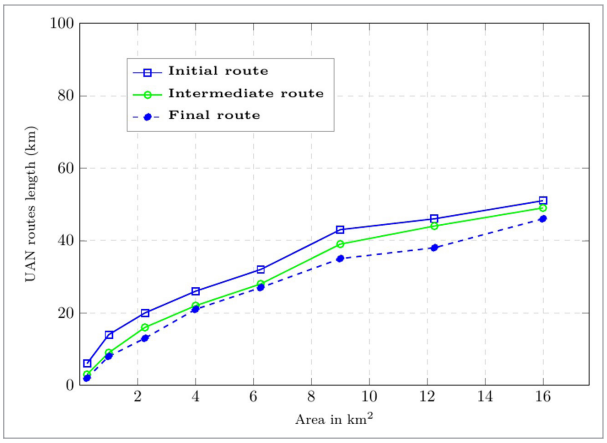


We note in passing, however, that the rate of route length increase starts to slow down after a certain threshold (≈12) which means that the solution is more effective in large areas than in small areas.

That is good since, after all, it is large areas that warrant the use of UAV anyway. The more important observation in the Figure is that, regardless of the area size, the route length produced by each of the three phases gets progressively shorter as we move to a higher phase, regardless of the area. That is why the uppermost curve represents the initial route and the lowermost represents the final route, with the middle curve being the intermediate route. This confirms that the three phases of the solution are essential to produce the shortest route for any given sensory area size. For any area size, it can be seen that for any area the initial route (uppermost) is the longest, followed by the intermediate route (middle), followed by the final route (lowermost).

The second experiment, whose results are shown in Figure 7, is similar to the first, except that the number of deployed SNs is smaller, namely 200 SNs. As such, the comments on the former figure still apply, but a couple of interesting observations here are worth mentioning. First, the length of all three routes increases with the sensing area size more slowly than was the case of 500 SNs. This is logical as when the number of SNs is small, increasing the deployment area size will not proportionally increase the hops between those SNs. We also notice that the difference in

Figure 7
The UAV routes length (km) in small and large scale area with N= 200 SNs

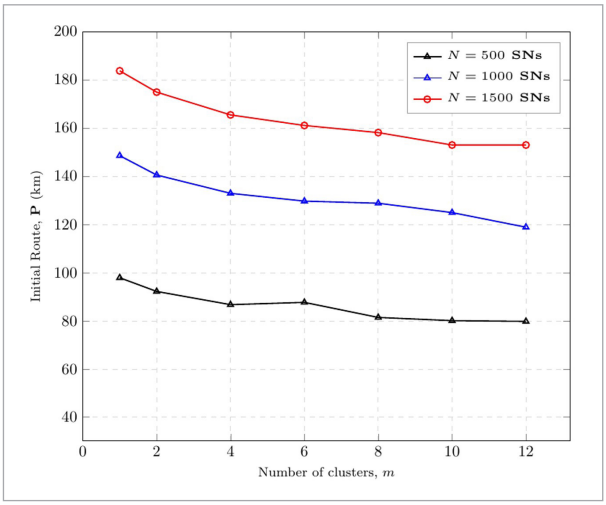


length between the three routes is not as significant as it was in the previous case of 500 SNs. This means that the proposed solution is more useful when there is a large number of SNs deployed.

Having verified that the three phases are successful in shortening the UAV route as progressively as intended, we move next to analyzing the impact of clustering, done at Phase I, on the performance of the solution. To this end we conducted the third experiment to inspect the initial route length for different numbers of clusters. This experiment holds great significance because, as we explained in the above two experiments, a short initial route results necessarily in a short intermediate route and hence a short final route.

Figure 8, shows how the number m of clusters affects the length of the initial route **P**, for various numbers of SNs deployed over the same square area of 4×4 km². The leftmost point on any of the three curves signifies the scenario of no clustering, which involves employing ACO on the complete set $\mathscr{S}$ of deployed SNs. As can be seen, this point corresponds to the longest initial route **P.** As the set is partitioned into an increasing number of clusters, the length of the route decreases correspondingly. This observation lends support to the proposition that the ACO algorithm performs more effectively when applied to a reduced number of nodes, thereby justifying the structure of the proposed solution. One should not, however, be encouraged to increase the number of clusters dramatically, as the length would not necessarily decrease proportionally, as can be seen from curves. On the other hand, when

Figure 8
The cost of the initial route, **P**, in km and area 4×4 km² with different values of cluster number m and sensor number N



examining the curves vertically and analyzing their response to changes in the number of nodes, it becomes apparent that a smaller number of sensory nodes within the sensory field results in a shorter initial route length. After all, the route connects the SNs, so when their number increases the length of the route increases as well. However, the increase is not proportional as we can see from the uppermost (N=1500 SNs) and lowermost (N = 500 SNs) curves. When the number of SNs is tripled, the length of the route almost doubled only, regardless of the number of clusters. This is understandable as the deployment area remains the same in both cases. We also have several more observations.

First, we will notice that, regardless of the number of deployed SNs, as the number of clusters increases, the length of the initial route decreases. For example, if we stop at the point where the number of clusters is 6, i.e. m = 6, we will find that the initial route is longest when there are N = 1500 SNs and shortest when there are N = 500 SNs. This is logical as the initial route traverses all existing SNs, so the more SNs the longer the traversal path.

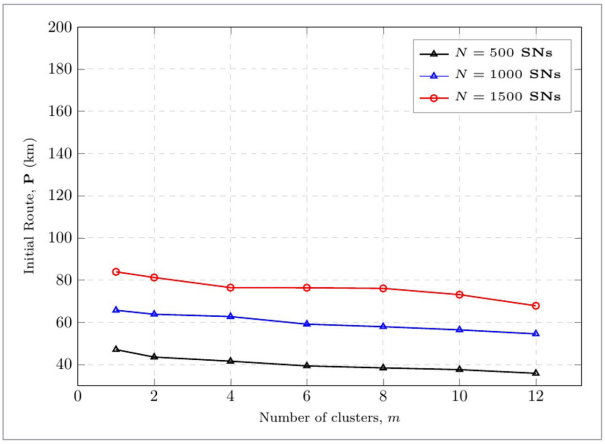
Second, for the same number N of SNs, the initial route gets shorter as the number of clusters gets larger. This again makes sense, as increasing the number of clusters, decreases the number of SNs within each cluster which in turn helps ACO to reach an optimal sub_route within the cluster. This effect seems more evident in the case when the number of SNs is large

than in the case when the number is small. For example, the slope of the uppermost curve is largest and that of the lowermost is smallest. The reason is that if the number of deployed SNs is small, then clustering will not bring down the number in each cluster too much. Conversely, more clustering of a large number of SNs will dramatically decrease the number within each cluster. That is clustering is more effective when the overall number of deployed SNs is large.

The fourth experiment, whose results are shown in Figure 9, is similar to the third, except that we have now a sensing area square of side 2 km. So, we expect similar observations, with some differences as follows. First, the length of the initial route here is smaller, for the same number of SNs. This is logical, since if we deployed the same number of SNs in two areas, one small and one large, the distances between the SNs would be longer in the latter case. Thus, regardless of any shortest path traversing obtained, the former would certainly be larger than the latter. Second, the rate of decrease of route length is also now smaller, regardless of the number N of SNs, as can be seen from the lower slopes of the three curves. This means it pays more to increase the number of clusters of the solution in a large sensory area than in a small sensory area, regardless of the number of SNs. This can be interpreted by observing that, for the same number of SNs, the distances between the SNs in a large area would be longer than in a small area. Thus, if a percentage decrease is exerted on both, it would be more noticeable in the large area. However, once again, one should not be encouraged to increase the number of clusters dramatically, as the route length would not necessarily decrease proportionally, as can be seen from the right half of the curves. Third, if we look at the curves vertically and focus on the behavior of the route length as the number of SNs changes, we will notice that the less SNs in the sensory field the shorter the length of the initial route. After all, the route connects the SNs, so when their number increases the length of the route increases as well.

Figure 9

The cost of the route, **P** within area 2×2 km^2, different values of cluster number m and sensor number N



By comparing Figures 8 and 9, we can see that the decrease in route length due to clustering is more pronounced when the sensing area gets larger. We note also that the decrease in length due to the number of SNs, regardless of the number of clusters, is more pronounced when the number of SNs gets larger.

Finally, to evaluate the performance of the proposed solution objectively, we ran a comparison experiment to test the proposed solution against four similar solutions recently published, namely in [56], [41], [49] and [29].

The experiment used a square sensory field of side 500 m, where 100 SNs were randomly deployed. The altitude of flight is 30 m and the SN transmission range is 50 m. The final route of the proposed approach is determined by considering these parameters. Four different scenarios are analyzed and their outcomes are averaged to obtain a comparative value. In the first scenario, the route is computed commencing from the leftmost SN within the designated region. Conversely, in the second scenario, the route is calculated from the rightmost SN in the same region. The third scenario involves computing the route starting from the highest SN within the said area. Finally, the fourth scenario computes the route starting from the lowest SN within the designated area. Table 7 shows the results and average of the four experiments.

Table 7

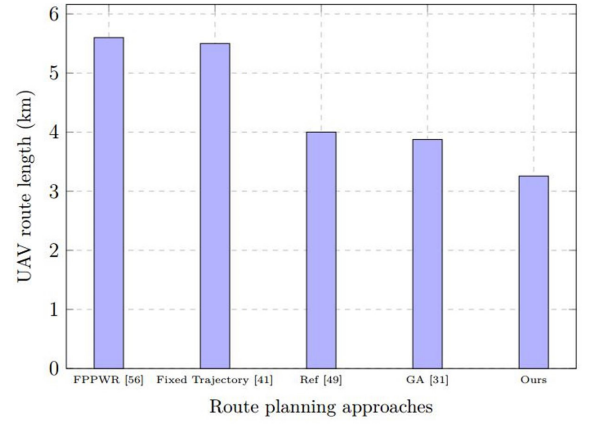
The results of final route calculation using four different scenarios

Start SN	Final route
left(km)	2.911975
right(km)	2.741445
above(km)	2.825351
below(km)	2.7728
Avg(km)	2.812892

The results of the experiment are shown in Figure 10, where it can be seen vividly that the proposed solution produces a DC route 19.28% shorter than the shortest route produced by the four competitive solutions. That is a great savings that is reflected not only on the energy consumption of the UAV, but also on the latency of the data.

Figure 10

The UAV route length within area 500×500 m^2, N = 100 and the SN transmission range 50 m



The energy consumption evaluation of the proposed solution is conducted using the simplest energy radio communication model in [31] on the part of the SN. In this model, SN transmitters consume energy for radio electronics and power amplifiers, while SN receivers dissipate energy on radio electronics. The energy consumption for transmitting b bits over a distance d_j from SN j to UAV is denoted as $E_{Tx}\left(b,d_j\right)$ and expressed as:

$$E_{Tx}\left(b,d_j\right) = \begin{cases} b\times\left(E_{elec}+\phi_{fs}\times d_j^2\right), d_j \prec d_{thr}. \\ b\times\left(E_{elec}+\phi_{mp}\times d_j^4\right), d_j \prec d_{thr} \end{cases} \quad (7)$$

where E_{elec} represents the energy dissipated per bit by the transmitter or receiver circuit. The coefficients of free space and multipath fading channel are denoted as Φ_{fs} and Φ_{mp}, respectively. The transmission distance threshold d_{thr} is given as

$$d_{thr} = \sqrt{\frac{\Phi_{fs}}{\Phi_{mp}}}. \quad (8)$$

The expended energy $E_{Rx}(b)$ to receive a b-bit packet is given as

$$E_{Rx}(b) = b\times E_{elec}. \quad (9)$$

Using Equations (7)-(9), the total energy consumption of the proposed solution is compared to five alternative methods, namely, LEACH-B [47], BPK-means [46], Park's approach [32], mk-means and NG-WSNs [22]. To ensure a fair comparison, we conducted an experiment using identical environmental parameter values as presented in [22]. In this experiment, a total of 100 SNs are deployed in a field with an area of 100×100 m^2, the data packet size is 3200 bits, E_{elec} = 50 nJ/bit, Φ_{fs} = 10 pJ/bit/m^2, Φ_{mp} = 0.0013 pJ/bit/m^4 and each SN is initially provided with an energy value of 1 joule. The UAV operates at an altitude of 25 m, while the transmission range of the SNs is set to 30 m.

Figure 11

Energy consumption comparison: Proposed solution vs. alternative methods

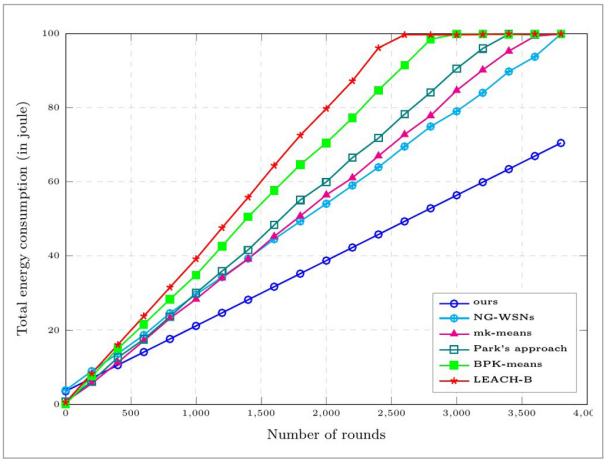


Figure 11 compares the total energy consumption across rounds. Initially, when the rounds are below 400, all methods exhibit similar energy usage. However, beyond 400 rounds, the alternative methods consume more power than the proposed solution. By the 3000-round mark, more than 99% of total energy

in the alternative methods is depleted, while the proposed solution retains 44% energy in most SNs, ensuring an extended network lifetime. This highlights the proposed solution's ability to keep more SNs operational for a longer duration compared to the alternative methods.

5. Conclusions

In this article we have presented a comprehensive solution to collect data from a large number of SNs deployed over a wide geographical area using a UAV. The solution is composed of three phases, each intended to output a route for the UAV shorter than that outputted by the previous phase. We use K-means in Phase I to partition the set of SNs into clusters. This has proved, from the experimental work carried out, fruitful in two respects. First, it helps the optimization algorithm, ACO, to reach an optimal solution, i.e. a shortest path traversing the SNs of the cluster. Second, it guarantees reaching this solution quickly. We develop an ingenious method to connect the shortest paths of the clusters together, obtaining an initial route that traverses all the SNs. In Phase II, we group the SNs along the initial route, with each group just fitting within the footprint of the UAV. The reason behind this is allow the UAV to collect the data of the entire group in one stop, hovering, above the centroid of the group. The collection of all the centroids forms the intermediate route. In Phase III, we apply again the optimization algorithm, ACO, to rearrange the centroids of the group to hopefully form a route that is shorter than the intermediate route. The rationale for this iterative application of ACO is that the distribution of the generated centroids has changed within the monitoring area. By reapplying ACO, we aim to construct a final route that is optimized for the updated distribution, ultimately yielding a route shorter than the intermediate one. The experimental work demonstrates that the solution works perfectly as intended. When its output, the final route, is compared with the output of four state of the art competitive solutions, it came out about 19.28% shorter than the shortest route of the four solutions. Finally, the proposed solution outperforms alternative competitive methods in terms of energy consumption and network lifetime.

References

1. Abdolreza, H., Salwani, A., Hossein, N. A Combined Approach for Clustering Based On K-Means and Gravitational Search Algorithms. Swarm and Evolutionary Computation, 2012, 6, 47-52. https://doi.org/10.1016/j.swevo.2012.02.003
2. Abu-Baker, A., Shakhatreh, H., Sawalmeh, A., Alenezi, A. H. Efficient Data Collection in UAV-Assisted Cluster-Based Wireless Sensor Networks for 3d Environment: Optimization Study. Journal of Sensors, 2023, 2023(1), 9513868. https://doi.org/10.1155/2023/9513868
3. Agarwal, V., Tapaswi, S., Chanak, P. Energy-Efficient Mobile Sink-Based Intelligent Data Routing Scheme for Wireless Sensor Networks. IEEE Sensors Journal, 2022, 22(10), 9881-9891. https://doi.org/10.1109/JSEN.2022.3164944
4. Ahn, H., Cho, H. J. Research of Multi-Object Detection and Tracking Using Machine Learning Based on Knowledge for Video Surveillance System. Personal and Ubiquitous Computing, 2022, 26(2), 385-394. https://doi.org/10.1007/s00779-019-01296-z
5. Albu-Salih, A. T., Seno, S. A. H. Energy-Efficient Data Gathering Framework-Based Clustering Via Multiple UAVs in Deadline-Based WSN Applications. IEEE Access, 2018, 6, 72275-72286. https://doi.org/10.1109/ACCESS.2018.2882161
6. Amar, M. A., Khaznaji, W., Horchani, L. Ptsp Solution Strategy for Motion Trajectory of UAV in Ubiquitous Sensor Network. Procedia Computer Science, 2020, 176, 3191-3199. https://doi.org/10.1016/j.procs.2020.09.130
7. Baek, J., Han, S. I., Han, Y. Energy-Efficient UAV Routing for Wireless Sensor Networks. IEEE Transactions on Vehicular Technology, 2019, 69(2), 1741-1750. https://doi.org/10.1109/TVT.2019.2959808
8. Bouhamed, O., Ghazzai, H., Besbes, H., Massoud, Y. A UAV-Assisted Data Collection for Wireless Sensor Networks: Autonomous Navigation and Scheduling. IEEE Access, 2020, 8, 110446-110460. https://doi.org/10.1109/ACCESS.2020.3002538
9. Cameron, P. J. Combinatorics: Topics, Techniques, Algorithms. Cambridge University Press, 1994. https://doi.org/10.1017/CBO9780511803888

10. Chang, C. Y., Chen, S. Y., Chang, I. H., Yu, G. J., Roy, D.S. Multirate Data Collection Using Mobile Sink in Wireless Sensor Networks. IEEE Sensors Journal, 2020, 20(14), 8173-8185. https://doi.org/10.1109/JSEN.2020.2981692

11. Chen, J., Tang, J. Uav-Assisted Data Collection for Dynamic and Heterogeneous Wireless Sensor Networks. IEEE Wireless Communications Letters, 2022, 11(6), 1288-1292. https://doi.org/10.1109/LWC.2022.3164784

12. Chen, M., Liang, W., Li, Y. Data Collection Maximization for UAV-Enabled Wireless Sensor Networks. IEEE 29th International Conference on Computer Communications and Networks (ICCCN), 2020, 1-9. https://doi.org/10.1109/ICCCN49398.2020.9209619

13. Chen, M., Liang, W., Das, S. K. Data Collection Utility Maximization in Wireless Sensor Networks Via Efficient Determination of UAV Hovering Locations. IEEE International Conference on Pervasive Computing and Communications (PerCom), 2021, 1-10. https://doi.org/10.1109/PERCOM50583.2021.9439126

14. Chen, M., Liang, W., Li, J. Energy-Efficient Data Collection Maximization for UAV-Assisted Wireless Sensor Networks. IEEE Wireless Communications and Networking Conference (WCNC), 2021, 1-7. https://doi.org/10.1109/WCNC49053.2021.9417258

15. Chowdhury, S., Roy, A., Benslimane, A., Giri, C. On Semantic Clustering and Adaptive Robust Regression Based Energy Aware Communication with True Outliers Detection in WSN. Ad Hoc Networks, 2019, 94, 101934. https://doi.org/10.1016/j.adhoc.2019.101934

16. Demiane, F., Sharafeddine, S., Farhat, O. An Optimized UAV Trajectory Planning for Localization in Disaster Scenarios. Computer Networks, 2020, 179, 107378. https://doi.org/10.1016/j.comnet.2020.107378

17. Dicandia, F. A., Fonseca, N. J. G., Bacco, M., Mugnaini, S., Genovesi, S. Space-Air-Ground Integrated 6G Wireless Communication Networks: A Review of Antenna Technologies and Application Scenarios. Sensors, 2022, 22(9), 3136. https://doi.org/10.3390/s22093136

18. Donta, P. K., Amgoth, T., Annavarapu, C. S. R. An Extended ACO-Based Mobile Sink Path Determination in Wireless Sensor Networks. Journal of Ambient Intelligence and Humanized Computing, 2021, 12(10), 8991-9006. https://doi.org/10.1007/s12652-020-02595-7

19. Fang, T., Yang, Y. Distributed Communication Protocol in Wireless Sensor Network Based on Internet of Things Technology. Wireless Personal Communications, 2022, 126(3), 2361-2377. https://doi.org/10.1007/s11277-021-09203-7

20. Fellah, K., Kechar, B. New Approach Based on Hilbert Curve for Energy Efficient Data Collection in WSN With Mobile Sink. IET Wireless Sensor Systems, 2020, 10(5), 214-220. https://doi.org/10.1049/iet-wss.2019.0078

21. Gang, T., Congqiang, T., Claramunt, C., Hu, X., Zhou, P. Geometric A-Star Algorithm: An Improved A-Star Algorithm for AGV Path Planning in A Port Environment. IEEE Access, 2021, 9, 59196-59210. https://doi.org/10.1109/ACCESS.2021.3070054

22. Haider, S. K., Jiang, A., Almogren, A., Rehman, A. U., Ahmed, A., Khan, W. U., Hamam, H. Energy Efficient UAV Flight Path Model for Cluster Head Selection in Next-Generation Wireless Sensor Networks. Sensors, 2021, 21(24), 8445. https://doi.org/10.3390/s21248445

23. Haoran, M., Khan, M. F., Peng, L., Tak, B., Lee, J., Ho, P.-H. Data-Similarity-Based IoT Node Selection for UAV Trajectory Optimization. Computers and Electrical Engineering, 2023, 112, 108994. https://doi.org/10.1016/j.compeleceng.2023.108994

24. Jiang, B., Huang, G., Wang, T., Gui, J., Zhu, X. Trust Based Energy Efficient Data Collection with Unmanned Aerial Vehicle in Edge Network. Transactions on Emerging Telecommunications Technologies, 2022, 33(6), e3942. https://doi.org/10.1002/ett.3942

25. Jin, H., Jin, X., Zhou, Y., Guo, P., Ren, J., Yao, J., Zhang, S. A Survey of Energy Efficient Methods for UAV Communication. Vehicular Communications, 2023, 41, 100594. https://doi.org/10.1016/j.vehcom.2023.100594

26. Katarzyna, P., Dawid, P., Gautam, S. Neuro-Heuristic Pallet Detection for Automated Guided Vehicle Navigation. IEEE International Conference on Big Data, Osaka, Japan, 2022, 6325-6331. https://doi.org/10.1109/BigData55660.2022.10020256

27. Krishnan, M., Lim, Y. Reinforcement Learning-Based Dynamic Routing Using Mobile Sink for Data Collection in WSNs and IoT Applications. Journal of Network and Computer Applications, 2021, 194, 103223. https://doi.org/10.1016/j.jnca.2021.103223

28. Lin, C., Han, G., Qi, X., Du, J., Xu, T., Martínez-García M. Energy-Optimal Data Collection for Unmanned Aerial Vehicle Aided Industrial Wireless Sensor Network-Based Agricultural Monitoring System: Clustering Compressed Sampling Approach. IEEE Transactions on Industrial Informatics, 2021, 17(6), 4411-4420. https://doi.org/10.1109/TII.2020.3027840

29. Liu, S., Wei, Z., Guo, Z., Yuan, X., Feng, Z. Performance Analysis of UAVs Assisted Data Collection in Wireless Sensor Network. IEEE 87th vehicular technol-

ogy conference (VTC Spring), 2018, 1-5. https://doi.org/10.1109/VTCSpring.2018.8417673

30. Liu, X., Liu, Y., Zhang, N., Wu, W., Liu, A. Optimizing Trajectory of Unmanned Aerial Vehicles for Efficient Data Acquisition: A Matrix Completion Approach. IEEE Internet of Things Journal, 2019, 6(2), 1829-1840. https://doi.org/10.1109/JIOT.2019.2894257
31. Li, Y., Liang, W., Xu, W., Jia, X. Data Collection of IoT Devices Using an Energy-Constrained UAV. IEEE International Parallel and Distributed Processing Symposium (IPDPS), 2020, 644-653. https://doi.org/10.1109/IPDPS47924.2020.00072
32. Luo, C., Chen, W., Li, D., Wang, Y., Du, H., Wu, L., Wu, W. Optimizing Flight Trajectory of UAV for Efficient Data Collection in Wireless Sensor Networks. Theoretical Computer Science, 2021, 853, 25-42. https://doi.org/10.1016/j.tcs.2020.05.019
33. Munan, L. Efficiency Improvement of Ant Colony Optimization in Solving the Moderate LTSP. Journal of Systems Engineering and Electronics, 2015, 26(6), 1300-1308. https://doi.org/10.1109/JSEE.2015.00142
34. Nawkhare, R., Singh, D. Machine Learning Approach on Efficient Routing Efficient Techniques in Wireless Sensor Network. IEEE International Conference on Current Development in Engineering and Technology (CCET), 2022, 1-6. https://doi.org/10.1109/CCET56606.2022.10080050
35. Nazib, R. A., Moh, S. Energy-Efficient and Fast Data Collection in UAV-Aided Wireless Sensor Networks for Hilly Terrains. IEEE Access, 2021, 9, 23168-23190. https://doi.org/10.1109/ACCESS.2021.3056701
36. Nguyen, M. T., Nguyen, C. V., Do, H. T., Hua, H. T., Tran, T. A., Nguyen, A. D., Ala, G., Viola, F. UAV-Assisted Data Collection in Wireless Sensor Networks: A Comprehensive Survey. Electronics, 2021, 10(21), 2603. https://doi.org/10.3390/electronics10212603
37. Połap, D. Neuro-Heuristic Analysis of Surveillance Video in A Centralized IoT System. ISA Transactions, 2023, 140, 0019-0578, 402-411. https://doi.org/10.1016/j.isatra.2023.05.024
38. Popescu, D., Dragana, C., Stoican, F., Ichim, L., Stamatescu, G. A Collaborative UAV-WSN Network for Monitoring Large Areas. Sensors, 2018, 18(12), 4202. https://doi.org/10.3390/s18124202
39. Popescu, D., Stoican, F., Stamatescu, G., Ichim, L., Dragana, C. Advanced UAV-WSN System for Intelligent Monitoring in Precision Agriculture. Sensors, 2020, 20(3), 817. https://doi.org/10.3390/s20030817
40. Qayyum, T., Trabelsi, Z., Malik, A., Hayawi, K. Trajectory Design For UAV-Based Data Collection Using Clustering Model in Smart Farming. Sensors, 2021, 22(1), 37. https://doi.org/10.3390/s22010037
41. Samir, M., Sharafeddine, S., Assi, C. M., Nguyen, T. M., Ghrayeb, A. Uav Trajectory Planning for Data Collection from Time-Constrained IoT Devices. IEEE Transactions on Wireless Communications, 2019, 19(1), 34-46. https://doi.org/10.1109/TWC.2019.2940447
42. Saxena, N., Gupta, N., Gupta, J., Sharma, D. K., Dev, K. Trajectory Optimization for the UAV Assisted Data Collection in Wireless Sensor Networks. Wireless Networks, 2022, 28(4), 1785-1796. https://doi.org/10.1007/s11276-022-02934-w
43. Seraj, R., Islam, S. M., Ahmed, M. The K-Means Algorithm: A Comprehensive Survey and Performance Evaluation. Electronics, 2020, 9(8), 1295. https://doi.org/10.3390/electronics9081295
44. Sergio, V., Ariza-Sentís, M., Valente, J. VineLiDAR: High-Resolution UAV-Lidar Vineyard Dataset Acquired Over Two Years in Northern Spain. Data in Brief, 2023, 51, 109686. https://doi.org/10.1016/j.dib.2023.109686
45. Skondras, E., Michalas, A., Vergados, D. D. Mobility Management on 5G Vehicular Cloud Computing Systems. Vehicular Communications, 2019, 16, 15-44. https://doi.org/10.1016/j.vehcom.2019.01.001
46. Sun, P., Boukerche, A. Performance Modeling and Analysis of A UAV Path Planning and Target Detection in A UAV Based Wireless Sensor Network. Computer Networks, 2018, 146, 217-231. https://doi.org/10.1016/j.comnet.2018.09.022
47. Tan, L., Gong, Y., Chen, G. A Balanced Parallel Clustering Protocol for Wireless Sensor Networks Using K-Means Techniques. Second International Conference on Sensor Technologies and Applications (Sensorcomm 2008), 2008, 300-305. https://doi.org/10.1109/SENSORCOMM.2008.45
48. Tong, M., Tang, M. Leach-b: An Improved Leach Protocol for Wireless Sensor Network. 6th International Conference on Wireless Communications Networking and Mobile Computing (WiCOM), 2010, 1-4. https://doi.org/10.1109/WICOM.2010.5601113
49. Ullah, Z., Al-Turjman, F., Moatasim, U., Mostarda, L., Gagliardi, R. UAVs Joint Optimization Problems and Machine Learning to Improve the 5G and Beyond Communication. Computer Networks, 2020, 182, 107478. https://doi.org/10.1016/j.comnet.2020.107478
50. Wang, C., Ma, F., Yan, J., De, D., Das, S. K. Efficient Aerial Data Collection with UAV in Large-Scale Wireless

Sensor Networks. International Journal of Distributed Sensor Networks, 2015, 11(11), 286080. https://doi.org/10.1155/2015/286080

51. Wang, X., Liu, X., Cheng, C. T., Deng, L., Chen, X., Xiao, F. A Joint User Scheduling and Trajectory Planning Data Collection Strategy for the UAV-Assisted WSN. IEEE Communications Letters, 2021, 25(7), 2333-2337. https://doi.org/10.1109/LCOMM.2021.3067898
52. Wen, W., Zhao, S., Shang, C., Chang, C. Y. Eapc: Energy-Aware Path Construction for Data Collection Using Mobile Sink in Wireless Sensor Networks. IEEE Sensors Journal, 2017, 18(2), 890-901. https://doi.org/10.1109/JSEN.2017.2773119
53. Wichmann, A., Korkmaz, T. Smooth Path Construction and Adjustment for Multiple Mobile Sinks in Wireless Sensor Networks. Computer Communications, 2015, 72, 93-106. https://doi.org/10.1016/j.comcom.2015.06.001
54. Wu, Q., Sun, P., Boukerche, A. Unmanned Aerial Vehicle-Assisted Energy-Efficient Data Collection Scheme for Sustainable Wireless Sensor Networks. Computer Networks, 2019, 165, 106927. https://doi.org/10.1016/j.comnet.2019.106927
55. Yang, D., Wu, Q., Zeng, Y., Zhang, R. Energy Tradeoff in Ground-to-UAV Communication Via Trajectory Design. IEEE Transactions on Vehicular Technology, 2018, 67(7), 6721-6726. https://doi.org/10.1109/TVT.2018.2816244
56. Yang, X. Introduction to Mathematical Optimization. From Linear Programming to Metaheuristics, 2008. https://doi.org/10.1007/s00521-016-2254-3
57. Yuan, J., Liu, W., Wang, J., Shi, J., Miao, L. An Efficient Framework for Data Aggregation in Smart Agriculture. Concurrency and Computation: Practice and Experience, 2021, 33(10), e6160. https://doi.org/10.1002/cpe.6160
58. Yue, W., Jiang, Z. Path Planning for UAV to Collect Sensors Data Based on Spiral Decomposition. Procedia Computer Science, 2018, 131, 873-879. https://doi.org/10.1016/j.procs.2018.04.291
59. Zeng, Y., Zhang, R., Lim, T. J. Wireless Communications with Unmanned Aerial Vehicles: Opportunities and Challenges. IEEE Communications Magazine, 2016, 54(5), 36-42. https://doi.org/10.1109/MCOM.2016.7470933
60. Zhu, B., Bedeer, E., Nguyen, H. H., Barton, R., Henry, J. UAV Trajectory Planning in Wireless Sensor Networks for Energy Consumption Minimization by Deep Reinforcement Learning. IEEE Transactions on Vehicular Technology, 2021, 70(9), 9540-9554. https://doi.org/10.1109/TVT.2021.3102161