Energy-Efficient UAV Trajectory Planning based on Flexible Segment Clustering Algorithm

Haoran Mei[1] and Limei Peng[1,*]
[1] School of Computer Science and Engineering, Kyungpook National University, Deagu, South Korea
Email: [1]{meihaoran, auroraplm}@knu.ac.kr
[*]Corresponding author

This paper plans the energy-efficient UAV trajectory when a UAV gathers data from massive IoT devices in a given area. The UAV trajectory design is addressed by two steps, i.e., IoT node clustering and UAV flight path planning for scanning the clusters, which are formulated as Cluster Minimization (CM) problem and Traveling Salesman Problem (TSP) in this work, respectively. The CM aims to contribute fewest clusters with minimal overlap to cover all the IoT devices and the per cluster size approaching the UAV communication coverage. On the other hand, the TSP seeks to design the shortest flight path to cover all the grouped clusters while minimizing energy consumption. Specifically, this work mainly focuses on the CM problem since the TSP issues have been well addressed in the past. In particular, we design a two-stage ILP optimization model to formulate the CM problem and propose two flexible clustering algorithms with low complexity, i.e., segment clustering (SC) and its variant, saying shifted SC (SSC). For the proposed ILP model and algorithms, we conduct extensive simulations under five different topologies and compare the performance results with existing methods. The simulation results indicate that the performance achieved by the proposed SSC algorithm is closest to the optimal results obtained from the ILP model. Moreover, it outperforms the existing methods under most topologies regarding cluster numbers, trajectory path length, and power consumption.

***Index Terms*—IoT, UAV, ILP, clustering, trajectory planning, data collection**

I. Introduction

In recent years, Unmanned Aerial Vehicles (UAVs) or drones have been widely used in diverse civil applications, taking their high mobility, low maintenance cost and ease of deployment into consideration [1]. Specifically, they are widely used to monitor the real-time transport traffic, remotely sense the extreme environment, provide wireless connectivity under natural disasters or malicious attacks, etc. Despite all benefits, UAVs' capability in aiding various applications is mainly restricted by its the flight endurance which is supported by its limited available power energy [1], [2].

Since power consumption is largely affected by drone propulsion, a feasible solution is to control the UAV movement carefully in order to avoid the unnecessary energy consumption due to redundant drone ascending and descending movements [3]. Besides, hovering and cruising in the designed trajectory also contribute to the UAV power consumption. These are then significantly affected by the number of hovering points, the path length of the trajectory, and the volume of data to collect.

Numerous studies focus on reducing the trajectory power consumption [3]–[10], with most of them dividing their solutions into the CM and TSP sub-issues. The existing TSP approach can be categorized into three classes: exact, approximate/heuristic and deep learning (DL) based. Exact algorithm involves Dynamic Programming algorithm and Integer Programming. For approximate/heuristic methodology, Google proposed OR-tools with different heuristics (e.g. Tabu Search and Simulated Annealing) for navigation in the search space and Local Search techniques for refining solution [11]. Although effective, the above methods are applied for static environments. To serve dynamic network, [4] addressed the online path planning through a double Q-learning network (DQN) based algorithm. [5], [12] employs transformer to solve TSP as a translation problem to convert the source language (i.e. a set of 2D points) to the target language (i.e. the point sequence with shortest length). To achieve an improved optimization upon DL, the heuristic methods such as weighted A* are applied.

On the other hand, the researchers have different views on whether to elect cluster headers (CHs) in CM. Specifically, some tend to elect a node in a cluster as a cluster header (CH), which is to launch data harvesting over all other cluster members and deliver them to the traversing UAVs [7]–[9], [12]. Nonetheless, the assumption that all IoT sensor nodes can communicate and collect data holds. Others ensure that UAVs simultaneously collect data from respective IoT devices in a cluster. For example, in [6], the authors employed the OFDMA technique to achieve this. Nonetheless, when serving massive IoT devices in the fixed UAV altitude as they assumed, it is difficult to avoid the considerable overlapping areas among clusters where UAVs repeatedly scan the same members and collect duplicate data. Selecting a specific set of IoT devices to collect data is non-trivial since it is hard to number all the devices and choose among them due to the sheer number of devices deployed [13].

Meanwhile, in [10], the authors proposed an approximation algorithm which can optimally pick up the hover-

Manuscript received May 20, 2023; revised September 26, 2023. Corresponding author: Limei Peng (email: auroraplm@knu.ac.kr).

ing positions to minimal the overlap, aiming to address a data harvesting utility maximization problem (UMP). Their algorithm, however, requires a considerably high time complexity, especially in the environment with the dense deployment of IoT devices. The low time-complexity method, namely the tiling method, has been proposed in [14] to cluster the IoT devices. It covers the plane with disjoint hexagons to minimize the overlapping areas and designate the centers of hexagons containing at least one IoT device as the hovering points. Some tiling-based variants, such as tiling method with shifting (TS) and tiling method with circular extension (TCE), have also been proposed to further optimize the cluster results. However, the empirical tests showed that those heuristics cannot always ensure a good tradeoff between the overlap and the path length.

Motivated by this, we propose concise algorithms which can find a smallest set of clusters to cover all given nodes with minimal overlapping areas in low complexity meanwhile help construct tours with acceptable length. We summary the major contributions of this work as follows. First, the trajectory designing puzzle is formulated as two sub-problems, saying cluster minimization (CM) and traveling salesman problem (TSP). Next, we show the NP-hardness of the CM puzzle based on Set Covering Optimization Problem (SCOP). In addition to proposing a two-stage integer linear programming (ILP) model, we also propose a low complexity algorithm and its variant, saying segment clustering (SC) and shifted segment clustering (SSC), respectively, to deal with the CM puzzle. Finally, the proposed approaches are compared with existing clustering techniques through simulation under five disparate topology scenarios.

The remainder of the paper is structured as follows. The system model, assumptions, power consumption model and problem statement are introduced in Section II. The details of the two-stage ILP model and the proposed SC and SSC algorithms for addressing the CM problem are presented in Section III. The simulation results are exhibited and analyzed in Section IV. Finally, we summary the paper in Section V.

II. Preliminaries

In this section, we present the details on the system model, assumptions, power consumption model, and problem statement.

A. System Model

The environment is considered as a $\mathscr{X} \times \mathscr{Y}$ m^2 region of interest with a set of N IoT, i.e., $D = \{d_1, d_2, ..., d_N\}$ as shown in fig. 1. These IoT nodes are sprinkled in the region and each one has a different coordinate, i.e, $d_i = (DX_i, DY_i)$, $1 \le i \le N$. For each tour, a fully charged UAV is dispatched to cruise and scan nodes of interest to achieve data collection tasks with a designed trajectory. Meanwhile, when the UAV moves to the proximity of an IoT node, the later can build a light-of-sight (LoS) link to deliver its sensory measurement to the former.

We assume that N IoT nodes can be grouped into M clusters, where the coverage size of each cluster is associated to the communication coverage of the UAV. We denote each cluster as a set of several IoT nodes, i.e. T_j, which is defined as follows.

$$T_j = \left\{ d_i \middle| ||d_i - c_j|| \le \gamma, d_i \in D \right\}, \forall c_j \in C \tag{1}$$

where $1 \le j \le M$ and $||a - b||$ calculates the Euclidean distance between two positions a and b. These clusters form a set $\boldsymbol{CS}$ = {T_1, T_2, T_3, ...}.

The UAV hovers at a location right above the center of each cluster to collect data from all nodes in the cluster at the same data rate. Such a location is designated as a way-point for the UAV, and the set of the way-points of the clusters is denoted by $C = \{c_1, c_2, c_3, ...\}$, where each element corresponds to a 2D hovering coordinate, i.e., $c_j = (CX_j, CY_j)$, where $1 \le j \le M$. The UAV trajectory is formed by $M+1$ hovering points as a closed path. We represent it by $\Theta = (\theta_0, \theta_1, ..., \theta_M, \theta_0)$, where the path both starts and ends θ_0 referring to the depot; $\theta_i = (CX_i, CY_i, Z_i)$ is a three dimensional hovering point, where the third coordinate Z_i refers to height. Here, (CX_i, CY_i), the first two dimensional coordinates in θ_i, are the same to that of the way-point of cluster i. The following equation is to calculate the path length L as the sum of the Euclidean distance of any two neighboring hovering points,

$$L = ||\theta_M - \theta_0|| + \sum_{k=0}^{M-1} ||\theta_{k+1} - \theta_k|| \tag{2}$$

For each flight, an UAV is assumed to have sufficient energy to visit each hovering point in the trajectory only once to achieve scanning the required area.

In this study, we consider the possibility that clusters will likely overlap with each other, and thus the UAV will collect replicated data from the nodes in the overlapping areas, leading to redundant but remarkable power consumption. This work aims to find an optimal clustering decision to minimize the overlapping areas and to construct an UAV flight trajectory with the optimal energy efficiency for $M+1$ hovering points.

B. Problem Statement

The main factors leading to UAV power consumption include hovering for data harvesting, trajectory travelling, and UAV state changes between hovering and moving, ascending/descending, etc. Let ρ_{hover}, ρ_{travel}, and ρ_{change} represent the volume of energy consumed by the UAV in joules per data unit in the hovering and data harvesting process; per length unit in the trajectory cruising, and UAV state shift, respectively. We denote energy consumption due to hovering, flying, and UAV state changes as E_{hover}, E_{travel}, and E_{change}, respectively. The following equation calculates the total UAV power consumption in joules,

$$E_{total} = E_{hover} + E_{travel} + E_{change} \tag{3}$$

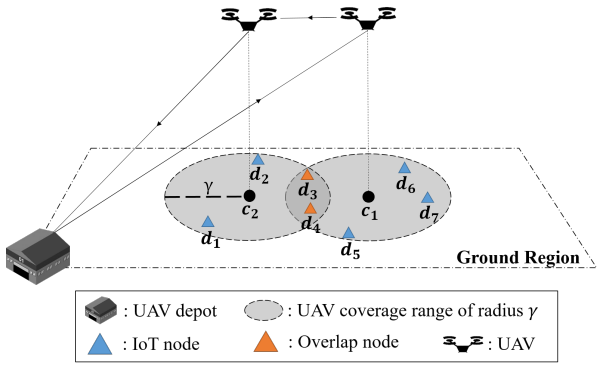


Fig. 1: The illustration of system model. γ refers to radius of the UAV communication coverage. Each cluster appears in shape of a circle, such as T_1 and T_2, where $T_1 = \{d_3, d_4, d_5, d_6, d_7\}$ and $T_2 = \{d_1, d_2, d_3, d_4\}$, respectively. c_1 and c_2 are two cluster central points, above where UAV hovers.

where

$$E_{hover} = \rho_{hover} \times \epsilon, \tag{4}$$

$$E_{travel} = \rho_{travel} \times L, \tag{5}$$

and

$$E_{change} = 2(M+1) \times \rho_{change} \tag{6}$$

with ϵ referring to the amount of collected data.

Based on the above definitions, the CM problem is defined as follows. Given a set $\mathscr{T} = \{T_1, T_2, T_3, ...\}$ of $\mathscr{M}$ candidate clusters where the center of each cluster serves as a candidate hovering coordinate $h_k = (HX_k, HY_k), 1 \le k \le \mathscr{M}$. Denote by $\mathscr{H} = \{h_1, h_2, h_3, ...\}$ the set of candidate hovering coordinates. The problem is to cover all the IoT nodes by minimal clusters with minimal overlap. The first stage aims at determining the minimum value M on the number of clusters and the second stage is to form the M clusters that can achieve minimal overlap. The CM problem is NP-hard and its NP-hardness can be converted to the Set Covering Optimization Problem (SCOP) [15].

C. NP-hardness of the Defined Problems

Theorem 1: The Cluster Minimization (CM) is NP-hard.

Proof: Denote an instance of SCOP and CM by $\mathbb{P}_{SCOP}$ and $\mathbb{P}_{CM}$, respectively. Given a finite set $\mathbb{D} = \{1, 2, 3, ...\}$ of N integers, a finite family $\mathbb{H}$ of $\mathscr{M}$ subsets of $\mathbb{D}$, SCOP aims to find a smallest collection $\mathscr{C} \subseteq \mathbb{H}$ whose union is $\mathbb{D}$.

Any SCOP instance can be reshaped as a CM instance as follows. The set $\mathbb{D}$ of N integers in $\mathbb{P}_{SCOP}$ corresponds to the set D of N IoT nodes in $\mathbb{P}_{CM}$ where each integer $i \in \mathbb{D}$ in $\mathbb{P}_{SCOP}$ can represent an IoT node $d_i \in D$ with $1 \le i \le N$. Then, the family $\mathbb{H}$ of $\mathscr{M}$ subsets of $\mathbb{D}$ in $\mathbb{P}_{SCOP}$ corresponds to the set $\mathscr{T}$ of $\mathscr{M}$ IoT cluster candidates in $\mathbb{P}_{CM}$. Thus, there always exists a subset of $\mathbb{D}$ in $\mathbb{H}$ in $\mathbb{P}_{SCOP}$ corresponding to each cluster candidate in $\mathbb{P}_{CM}$, and the primary objective of $\mathbb{P}_{CM}$ is equivalent to that of SCOP in finding the fewest sets whose union is $\mathbb{D}$.

The solution of $\mathbb{P}_{CM}$ is to form a set $CS = \{T_1, T_2, ...\}$ of M IoT node clusters with full coverage and minimized overlap. On the other hand, there is a collection $\mathscr{C}$ of subsets of $\mathbb{D}$ corresponding to CS. Specifically, if $\bigcup_{T_j \in CS} T_j \neq D$, which means the clusters in CS cannot fully cover all the IoT nodes in set D, then $\mathscr{C}$ is not a feasible solution for $\mathbb{P}_{SCOP}$ since the union of $\mathscr{C}$ is not $\mathbb{D}$. Otherwise, $\mathscr{C}$ can serve as a feasible solution to $\mathbb{P}_{SCOP}$. Since a solution to $\mathbb{P}_{CM}$ can always correspond to a feasible solution to $\mathbb{P}_{SCOP}$, we can conclude that the solution is polynomial, and the theorem thus holds.

III. MATHEMATICAL MODEL AND ALGORITHM FOR CLUSTER MINIMIZATION PROBLEM

This section first formulates the CM problem into a two-stage ILP model and then introduces a low-complexity algorithm, i.e., Segment Clustering (SC), and its variant, i.e., shifted SC (SSC).

A. ILP Model

The first stage of ILP model is to identify the minimum value M on the cluster number. Assume that a scanning area consisting of a set D of IoT nodes is represented by a xy-plane divided into grids. We then assign a disk with radius γ at the center point of each grid, where γ is the UAV communication radius, trying to ensure that each grid is the quadrilateral of the disk. Note that each disk corresponds to a cluster. Based on this, we consider the following parameters to develop the ILP model.

- $B_{i,j}$: a binary parameter that is one if cluster $j \in \mathscr{T}$ covers the IoT node $i \in D$; zero otherwise.
- $\mathbb{F}$: a large number.

In addition, the variables are defined as follows:

- $\alpha_{i,j}$: a dummy variable that is one if IoT nodes i is grouped in cluster $j \in \mathscr{T}$; zero otherwise.
- β_j: a dummy variable that is one if cluster $j \in \mathscr{T}$ is finally selected; zero otherwise.

The ILP objective is to minimize the hovering points, which is converted to minimize the number of clusters,

$$\min \ M = \sum_{j \in \mathscr{T}} \beta_j \tag{7}$$

$$s.t. \quad \sum_{j \in \mathscr{T}} \alpha_{i,j} \ge 1, \ \forall i \in D \tag{8}$$

$$\alpha_{i,j} \le \beta_j, \ \forall i \in D, j \in \mathscr{T} \tag{9}$$

$$\alpha_{i,j} \le B_{i,j}, \ \forall i \in D, j \in \mathscr{T} \tag{10}$$

where eq. (7) refers to the primary objective of CM problem, i.e., the minimization of the number of the required clusters; constraint (8) implies that each IoT node is covered by at least one cluster; constraint (9) indicates that each selected cluster should cover at least one IoT node; constraint (10) ensures that an IoT node belongs to the cluster j only if it is covered by the cluster.

Based on the outcome M in the first stage, the following ILP model aims to Search M clusters with minimal overlap, meaning to cover the minimal number of IoT nodes, including the repeatedly covered ones, in the M clusters. The objective is represented by eq. (11).

$$\min \sum_{i\in D}\sum_{j\in\mathcal{T}} \alpha_{i,j} \tag{11}$$

$s.t.$ (5),(6),(7)

$$\sum_{j\in\mathcal{T}} \beta_j = M, \tag{12}$$

$$\alpha_{i,j} \geq \beta_j B_{i,j},\ \forall i \in D, j \in \mathcal{T} \tag{13}$$

where constraint (12) ensures that the number of selected clusters is exactly M and constraint (13) is to ensure that if node i is covered by cluster j, cluster j must be selected to form the final solution.

Through the above two-stage ILP model, the sets C and CS can be obtained.

B. Segment Clustering Algorithm

To date, there have been extensive studies on clustering algorithms involving K-Means++, BIRCH [16], DBSCAN [17], etc. Nonetheless, these methods either cannot perform the UAV communication range-based clustering or require high complexity. Motivated by this, a clustering algorithm with low-complexity considering the UAV communication range, namely segment clustering (SC), is proposed in this section.

The scanning area is represented by a XY-plane. The basic idea of SC is to split the scanning area into different sized grids, where the maximum size is less than the UAV communication coverage. The next step is to select minimal grids from those with high node densities to cover all IoT nodes and ensure the minimization of overlapping areas. The selected grids become the candidates for deploying clusters, and the center of each one becomes a way-point where a UAV hovers at for data gathering.

The following pseudo-codes illustrate the process of the proposed SC algorithm. The input parameters include the long edge of a rectangle, the radius of UAV communication coverage and the threshold on the cluster density, which are represented by s, γ, and TH_d, respectively. It is worth noting that assume the UAV communication pattern is represented by a circle, the inscribed square of the communication circle is the maximum area of a cell, and thus $s \leq \sqrt{2}\gamma$.

Figs. 2(b) and 2(c) show the first step of SC algorithm, where it segments the given areas based on the IoT node locations horizontally and then vertically. The details are illustrated in Pseudocode 1. After horizontal and vertical segment, there are rectangles with various sizes in Fig. 2(d). Each rectangle represents a grid cell and the set of grid cells is denoted by S_{cell}. The long edge of a grid should be less than or equal to s. Then, Fig. 2(e) shows that some grid cells are removed since their densities are lower than the threshold TH_d. Next, in Figs. 2(f), the remaining grid cells are replaced with circles, and each circle represents a cluster. Fig. 2(g) shows that, for the IoT nodes covered by the eliminated cells, some of them are reassigned to one of remaining clusters, and the others form a new cluster. The process repeats until each IoT node is covered by at least one cluster. The set of the generated clusters is denoted by CS while the center coordinates of those clusters form the set C.

Pseudocode 2 illustrates the rest part of SC. Particularly,

Pseudocode 1: SC1.1-Segmentation and clustering

Input: D, s, γ, TH_d
Output: The set formed by the selected hovering locations, C
Function *SectionGeneration(R, s)* **do**
  Generate a empty set for bounds of the sections on x-axis or y-axis, $B \leftarrow \emptyset$;
  Sort R in decreasing order.
  $r_{upper} \leftarrow R.pop()$;
  $r_{lower} \leftarrow r_{upper}$;
  while *r is not empty* **do**
    $r \leftarrow R.pop()$;
    if $r_{upper} - r \leq s$ **then**
      $r_{lower} \leftarrow r$
    else
      $B \leftarrow B \cup \{(r_{lower}, r_{upper})\}$;
      $r_{upper} \leftarrow r$;
      $r_{lower} \leftarrow r$;
    end
  end
  return B;
end
Generate a empty set for selected hovering coordinates, $C \leftarrow \emptyset$;
Generate a empty set for candidate hovering coordinates, $H \leftarrow \emptyset$;
Generate a empty set for IoT clusters, $CS \leftarrow \emptyset$;
Label all IoT nodes in D as ungrouped ones, $W \leftarrow D$;
Step I (Cell segmentation & update):
Generate vertical and horizontal sections, i,e., B_v and B_h:
  $B_v \leftarrow SectionGeneration(X_{node}, s)$;
  $B_h \leftarrow SectionGeneration(Y_{node}, s)$;
Generate cells based on sections. $B_{cell} \leftarrow B_v \times B_h$;
for *each cell* $k \in S_{cell}$ **do**
  Calculate the density of cell k, q_k as the number of IoT nodes within it;
  if $q_k == 0$ **then**
    $S_{cell} \leftarrow S_{cell} - \{k\}$;
  end
end
for *each cell* $k \in S_{cell}$ **do**
  if $q_k \geq TH_d$ **then**
    $H \leftarrow H \cup$ {the 2D center coordinate of the cell k};
  end
end
Step II (Clustering based on cells):
repeat
  $h^* \leftarrow \arg\max_{h_k \in H} \left|\{d_i : d_i \in W, ||d_i - h_k|| \leq \gamma\}\right|$;
  if $\left|\{d_i : d_i \in W, ||d_i - h^*|| \leq \gamma\}\right| == 0$ **then**
    Break;
  end
  $H \leftarrow H - \{h^*\}$;
  $C \leftarrow C \cup \{h^*\}$;
  $W \leftarrow W - \{d_i : d_i \in W, ||d_i - h^*|| \leq \gamma\}$;
  $CS \leftarrow CS \cup \{T_l = \{d_i : d_i \in D, ||d_i - h^*|| \leq \gamma\}\}$;
until $U == \emptyset \ \vee H == \emptyset$;
while $W \neq \emptyset$ **do**
  $d^* \leftarrow \arg\max_{d_j \in U} \left|\{d_i : d_i \in U, ||d_i - d_j|| \leq \gamma\}\right|$;
  $C \leftarrow C \cup \{d^*\}$;
  $W \leftarrow W - \{d_i : d_i \in W, ||d_i, d^*|| \leq \gamma^2\}$;
  $CS \leftarrow CS \cup \{T_l = \{d_i : d_i \in D, ||d_i - d^*|| \leq \gamma\}\}$;
end

Pseudocode 2: SC1.2-Overlap minimization

repeat
  $P \leftarrow \{(c_i, c_j) : ||c_i - c_j|| \leq \gamma, 1 \leq |T_i \cap T_j|, c_i, c_j \in NC, T_i, T_j \in CS\}$;
  if $P == \emptyset$ **then**
    Return C;
  else
    for *each* (c_i, c_j) *in* P **do**
      $DT \leftarrow T_i \cap T_j$;
      if $T_i \leq T_j$ **then**
        $T_j \leftarrow T_j \cup DT$;
        $T_i \leftarrow T_i - DT$;
      else
        $T_i \leftarrow T_i \cup DT$;
        $T_j \leftarrow T_j - DT$;
      end
    end
  end
  $C \leftarrow C - \{c_j | T_j = \emptyset, c_j \in C, T_j \in CS\}$;
  $CS \leftarrow CS - \{T_j | T_j = \emptyset, T_j \in CS\}$;
  for *each* $c_j \in NC$ **do**
    Update c_j via equation (14);
    Update T_j via equation (1);
  end
until *the positions in* C *remain unchanged*;

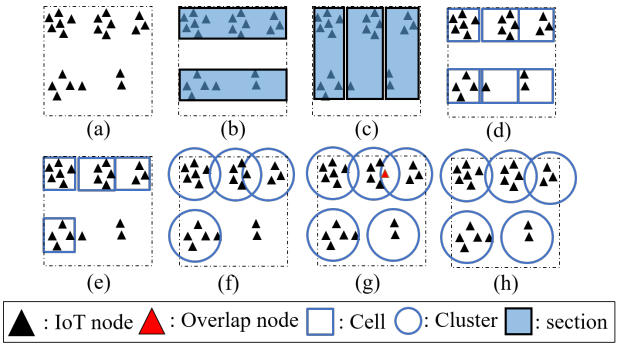


Fig. 2: The XY-plane segmentation

as shown in Fig. 2(h), this part is to implement the minimization of the overlap areas caused by clusters. If the distance two clusters is less than γ, then they can be defined as the adjacent clusters. If there are any IoT nodes covered by two adjacent clusters, SC will re-assign those nodes to the one with higher density. After this, the corresponding coordinates in C, $c_k \in C$ are updated by the the following equation:

$$c_k = \left(\frac{\sum_{d_i \in T_k} DX_i}{|T_k|}, \frac{\sum_{d_i \in T_k} DY_i}{|T_k|}\right) \tag{14}$$

where $|T_k|$ stands for the number of IoT nodes belonging to the cluster with coordinate c_k. The equation (1) is adopted to update the corresponding cluster. The repetition of this process is halted when there is no change to the clusters after updating.

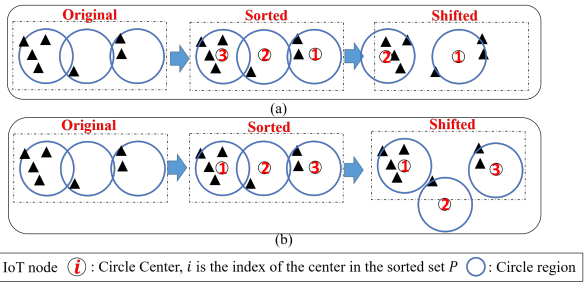


Fig. 3: Shift results for different sort configurations

C. Shifted Segment Clustering Algorithm

Since the SC algorithm determines the location of a hovering point based on the arithmetic mean of IoT node coordinates of a cluster, the IoT nodes are less likely to locate at the boundary of clusters. Hence, we can further reduce the overlapping region among clusters by shifting the hovering points until at least one IoT node locates at the boundary of a cluster. Regarding this, we propose a variant of SC based on iterative shift, namely Shifted Segment Clustering (SSC) algorithm.

The initial steps of SSC are the same as SC. It first segments the XY-plane to create a set of centers for clusters containing at least one IoT node, denoted by H, as shown in Pseudocode 1. Next, clusters with their centers in H are assigned a radius no larger than γ, limited by the UAV communication coverage. The additional steps of SSC, compared to SC, lie in that it sorts the centers in H and then iteratively shifts the centers to reduce the overlapping area and the possible redundant clusters.

Specifically, SSC first sorts the cluster centers based on their X-axis first and then the Y-axis (or Y-axis first and then X-axis) in the ascending or descending orders and then number the centers as shown in Fig. 3. The numbering sequence decides the shifting order of clusters, and a smaller number indicates a higher shifting priority. Note that based on whether X-axis or Y-axis is sorted first and the sorting order, the numbering may be different even for the same clusters as shown in Figs. 3(a) and 3(b), which would affect the shifting results.

Then SSC shifts the cluster centers following the numbering obtained after sorting. For a specific cluster, SSC tries to shift its center vertically and horizontally, aiming to detach it with the other clusters while including as many IoT nodes in it as possible until at least one of its original IoT node sites on its region boundary. By doing this, the overlap among clusters can be minimized and the cluster density can be maximized. Besides, the shifting process complies with the following rules. First, a cluster shifted previously can cover the IoT nodes of a cluster shifted later, but the inverse is not true. Therefore, the IoT nodes in a cluster being shifted may alter with each movement. Second, the IoT nodes currently in a cluster midst a shifting process should still be in the same cluster but possibly at the cluster boundary. Third, if a cluster does not cover any node due to its IoT nodes being re-grouped

Pseudocode 3: SCC1.1-Shifting and clustering

Input: D, s, γ, TH_d, OP
Output: The set formed by the selected hovering locations, C

Perform the same process from line 1 to line 35 in Pseudocode 1;
Step II (Shifting clustering):
$min_{overlap} \leftarrow$ a large integer number;
for *each sort option* OP_{sort} *in* OP **do**
  for *each shift option* OP_{shift} *in* OP **do**
    $H_{new} \leftarrow H$.sort(OP_{sort});
    $W \leftarrow D$;
    $CS_{new} \leftarrow \emptyset$;
    for *each* h *in* H_{new} **do**
      $W \leftarrow \{d_i : ||d_i - h|| \leq \gamma, d_i \in U\}$;
      if W *is empty OR* T *is empty* **then**
        $H_{new} \leftarrow H_{new} - p$;
      else
        shift h by following the option OP_{shift} and the shift length is calculated by Eq.(15);
        $W \leftarrow \{d_i : ||d_i - h|| \leq \gamma, d_i \in U\}$;
        $CS \leftarrow CS \cup \{T\}$;
        $W \leftarrow W - T$;
      end
    end
    $N_{overlap} \leftarrow \sum_{h \in H_{new}} |\{d_i : ||d_i - h|| \leq \gamma, d_i \in D\}| - |D|$;
    if $N_{overlap} \leq min_{overlap}$ **then**
      $min_{overlap} \leftarrow N_{overlap}$;
      $CS \leftarrow CS_{new}$;
      $C \leftarrow H_{new}$;
    end
  end
end

to previously shifted clusters, this cluster is removed. We can see that after shifting, the cluster numbered three is eliminated, and all clusters are detached with no overlap, as shown in Figs. 3(a) and 3(b), respectively.

We define the shift length of a center as the minimum value of the distances between the IoT nodes in a cluster and the cluster region boundary, denoted by ΔL_{h_j},

$$\Delta L_{h_j} = \min_{i \in IN_j} \Delta l_{ij}, \forall h_j \in H \tag{15}$$

where Δl_{ij} refers to the distance between IoT node i in the cluster with center h_j and the corresponding region boundary. The value of Δl_{ij} varies with the shifting directions and can be formulated as follows,

$$\Delta l_{ij} = \begin{cases} \sqrt{(\gamma)^2 - (\Delta x_{ij})^2} + b\Delta y_{ij}, & \text{if } h_j \in H \text{ shifts vertically} \\ \sqrt{(\gamma)^2 - (\Delta y_{ij})^2} + b\Delta x_{ij}, & \text{otherwise} \end{cases} \tag{16}$$

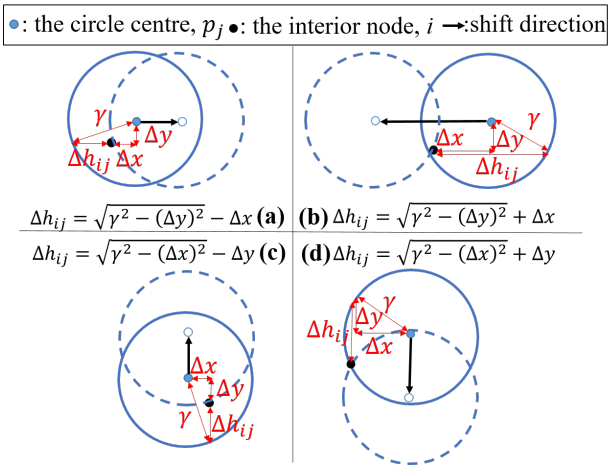


Fig. 4: The calculation of Δh_i and the translation of a centre.

where Δx_{ij} and Δy_{ij} are the absolute values of horizontal and vertical difference between IoT node i and its cluster center h_j, respectively; b is a coefficient varying with the shifting directions, which is one if the shifting direction is towards IoT node i; -1 otherwise, as illustrated by Fig. 4. Fig. 4 also describes the expected shifting results in case that there is only one IoT node covered by the cluster with center h_j, implying $\Delta L_{h_j} = \Delta l_i$.

IV. Numerical Analysis

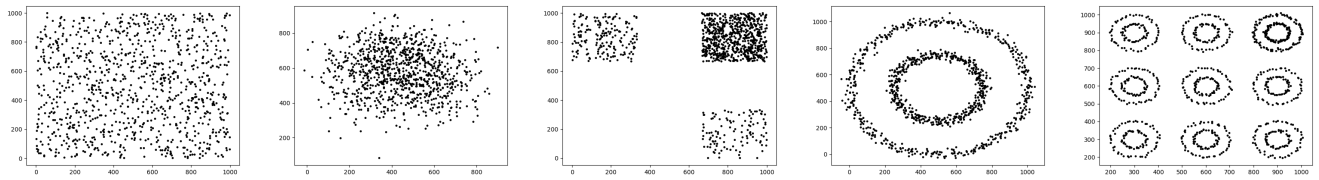
This section evaluates the performance of the proposed methods, saying SC, SSC, and ILP model and compares them with their counterparts, i.e., the approximation algorithm for UMP [10], TS, and TCE [14]. The UMP algorithm can achieve the minimization of the overlap areas but suffers from a considerable time complexity due to identifying many hovering candidates with each corresponding to a cluster with high density. TS and TCE first segment the XY-plane by tiling polygonal of the same size to roughly determine the hovering points and then reduce the number of hovering points through different techniques. However, the empirical tests show that using tiles of the same size leads to longer path lengths than using cells of various sizes.

A. Simulation Configuration

TABLE I: Simulation configuration for a UAV

Parameters	Values
Reception range	0~100 meter
Weight	1375g [18]
Energy capacity	321,206 J (for one battery) [18]
ρ_{travel}	22.9 J/m (at 5.56 m/s) [19]
ρ_{hover}	1.852 J/Mbit (at 103.2 Mbps)
ρ_{change}	50 J/status change

Here, Fig. 5 shows five different IoT nodes deployment scenarios, saying "uniform," "corner," "blobs," "ring," and



(a) uniform (b) blobs (c) corner (d) ring (e) uniform rings

Fig. 5: Five topology scenarios for the distribution of IoT nodes over the area with the size of 1000×1000 m^2

"uniform ring," respectively, which are considered for the feasibility estimation of the proposed algorithms. In each scenario, there are 1000 IoT nodes deployed over a XY-plane with the size of 1000×1000 m^2. The data rate of all IoT nodes is fixed at 103.2 Mbps. There are two configurations for the data sizes from IoT sensor nodes: 5 Mbits and 50 Mbits. The configuration of the drone model using DJI Phantom 4 Pro [18] is shown in Table I.

We implement The ILP models through LINGO. The proposed clustering algorithms, i.e., SC and SSC, and their counterparts, i.e., UMP, TS, and TCE, are implemented via Python 3.6. To solve the next TSP issue, which aims to contribute a closed path with the shortest length for a set of hovering locations/clusters obtained in the previous clustering step, we equally use the existing vehicle routing solver of Google OR Tools (ORT) [11] for all the clustering algorithms, i.e., SC, SSC, UMP, TS, and TCM.

B. Evaluation Metrics

To investigate the proposed algorithms, we first compare them with other counterparts based on criteria such as the number of clusters, repeated data ratio, and path length. The latter is considered the main factor impacting energy consumption. Next, to evaluate and compare the power consumption of the proposed algorithms, we define a metric as the relative difference between $E_{total}^{(SSC)}$ and $E_{total}^{(z)}$, where $E_{total}^{(SSC)}$ is the energy required by SCC algorithm and $E_{total}^{(z)}$ can be the total energy consumption for any algorithm except SSC (i.e. z can be UMP, SC, TS and TCE). The calculation formula of relative difference is as follows

$$\delta_{z,SSC} = \frac{E_{total}^{(z)} - E_{total}^{(SSC)}}{E_{total}^{(SSC)}} * 100\% \tag{17}$$

where $\delta_{z,SSC}$ refers to the relative difference between the energy consumption of SSC and the algorithm z. This metric converts values measured on different scales to a notionally common scale. Notably, if the energy consumption of SCC is less than that of the algorithm z, then $\delta > 0$; Otherwise, the $\delta \leq 0$. The degree of difference is reflected by the value of $|\delta|$.

Furthermore, we analysis the main components of energy consumption, i.e., hovering, flying, and UAV state changes given by the equations (4), (5), and (6), respectively, under different data size configuration, namely 1 Mbits, 50 Mbits and 500 Mbits.

C. Numerical Result

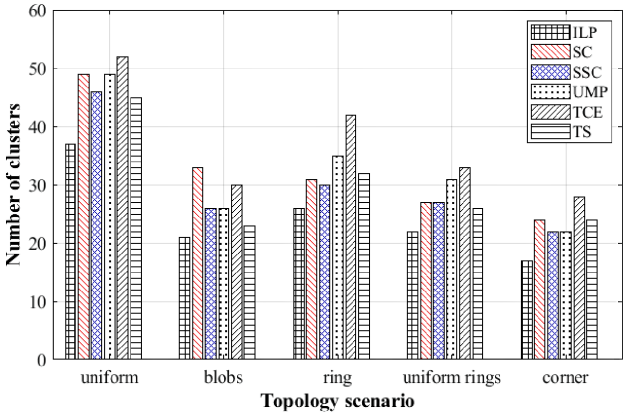


Fig. 6: Number of clusters

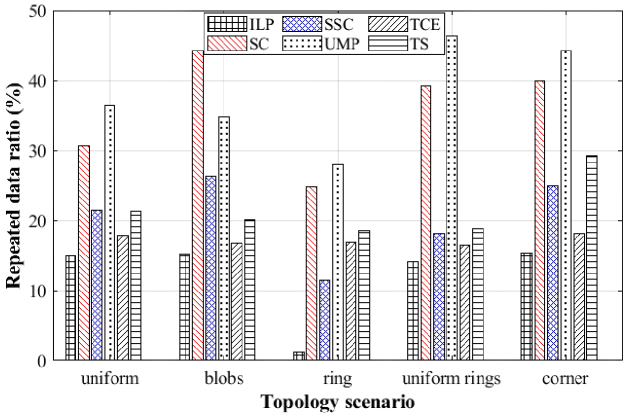


Fig. 7: Repeated data ratio

Table II illustrates the time complexities of the clustering algorithms in terms of their execution time. It can be observed that except for the optimization ILP model, which requires high execution time as expected to find the optimal solution, the proposed SC is the most time-efficient since it requires the least execution time, followed by TCE, TS, SSC, and UMP. Even though the proposed SSC requires slightly more execution time than TS and TCE, they are still in the same magnitude. In comparison, the time complexity of UMP is almost five orders higher than other methods due to the iterative process needed in UMP to adjust the threshold for clusters, which is time-consuming.

TABLE II: Elapsed time for different clustering methods in unit of second.

	ILP	SC	SSC	UMP	TCE	TS
uniform	3000s	0.078125	0.9375	3000s + 154.20	0.171875	0.546875
blobs	86	0.046875	0.953125	3000s + 252.89	0.125	0.375
ring	42.22	0.0625	0.90625	3000s + 110.28	0.125	0.46875
uniform rings	42.16	0.03125	0.890625	3000s + 139.80	0.125	0.4375
corner	41.88	0.046	0.875	3000s + 245.75	0.125	0.421875

*3000s:The time cost of finding the optimal solution by the solver exceeds 3000s for ever instance.
*$t_1 + t_2$:For UMP, the sum of the execution time of $\mathbb{M}$ set generation process, t_1 and that of IoT node clustering process, t_1.

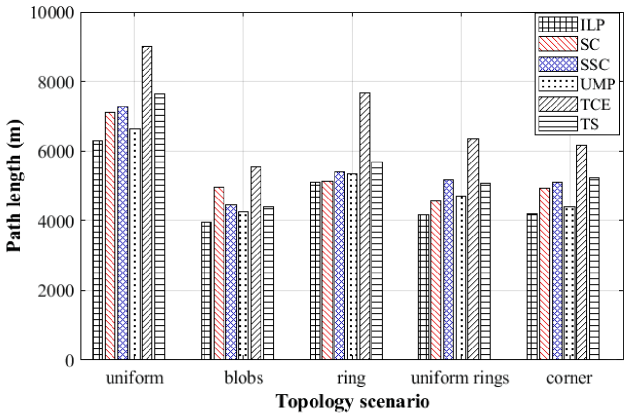


Fig. 8: Trajectory path length of different clustering techniques

Figs. 6 and 7 show the performance of different methods in terms of required clusters and data repetition ratios across five topology scenarios. The ILP model performs the best due to its exact nature. Thus, the performance of an algorithm close to the ILP model indicates its efficiency. The proposed SSC algorithm excels in the ring and corner scenarios, requiring the fewest clusters, and is the second-best in other scenarios, where TS performs the best. This is because decentralized and dense node deployment in certain scenarios allows SSC to achieve efficient clustering via minimal cells, further improved by the shift process. TCE collects the least repeated data in most scenarios, except for the ring scenario, where SSC performs the best, as seen in Fig. 7. Compared to SC, SSC significantly reduces data repetition ratios, including the shift process.

Fig. 8 shows simulation results for trajectory path lengths, determined by ORTool for hovering locations derived from grouped clusters. SC performs best in the ring and uniform rings scenarios, while UMP is superior in the other scenarios. However, UMP comes with significantly higher time complexity, as mentioned earlier. Notably, for the ring scenario, SC performs almost as well as the ILP optimization model. The shift process in SSC, aimed at eliminating overlaps, may lead to longer tour lengths.

The energy consumption performance was evaluated for different data size configurations (1 Mbits to 500 Mbits) across five scenarios (see Figs. 9a to 9e). Fig. 10 illustrates energy consumption due to hovering, flying, and state changes for data sizes of 5 Mbits, 50 Mbits, and 500 Mbits per IoT node. Overall, SSC has the lowest energy consumption in most scenarios (except blobs and corners) for data sizes between 50 Mbits and 400 Mbits. It's worth noting that SSC is the optimal choice for data sizes between 50 Mbits and 100 Mbits in most scenarios (except blobs). For scenarios with small data sizes, UAV energy consumption is mainly due to cruising, while hovering becomes the dominant factor as data sizes increase. In blobs scenario with large data sizes, TCE and TS outperform SSC in terms of the repeated data ratio while UMP wins in terms of path length for corner scenario with small data sizes. The impact of state changes on total energy consumption is negligible compared to other factors. Despite limitations in certain scenarios, SSC's performance in uniform scenario makes it a practical choice considering the prevalence of uniform scenario in the real world.

V. Conclusion

This paper proposed two-stages ILP model as well as a novel segment clustering (SC) algorithm and its variant shifted segment clustering (SSC) algorithm to address the clustering issue for UAV trajectory. The objective of both two techniques is the overlapping ratio minimization. Notably, compared to the existing clustering algorithms such as UMP, TCE, and TS, SC can take required significantly less time complexity while SSC can provide a better trade-off between the overlap and path length. The simulation results showed that ILP model performs the best. It also showed the high compatibility of SSC in various situations, since it outperforms its counterparts in most of the topology scenarios. Although effective, the proposed SSC might lead to a longer path length, which will be discuss in future work.

Acknowledgment

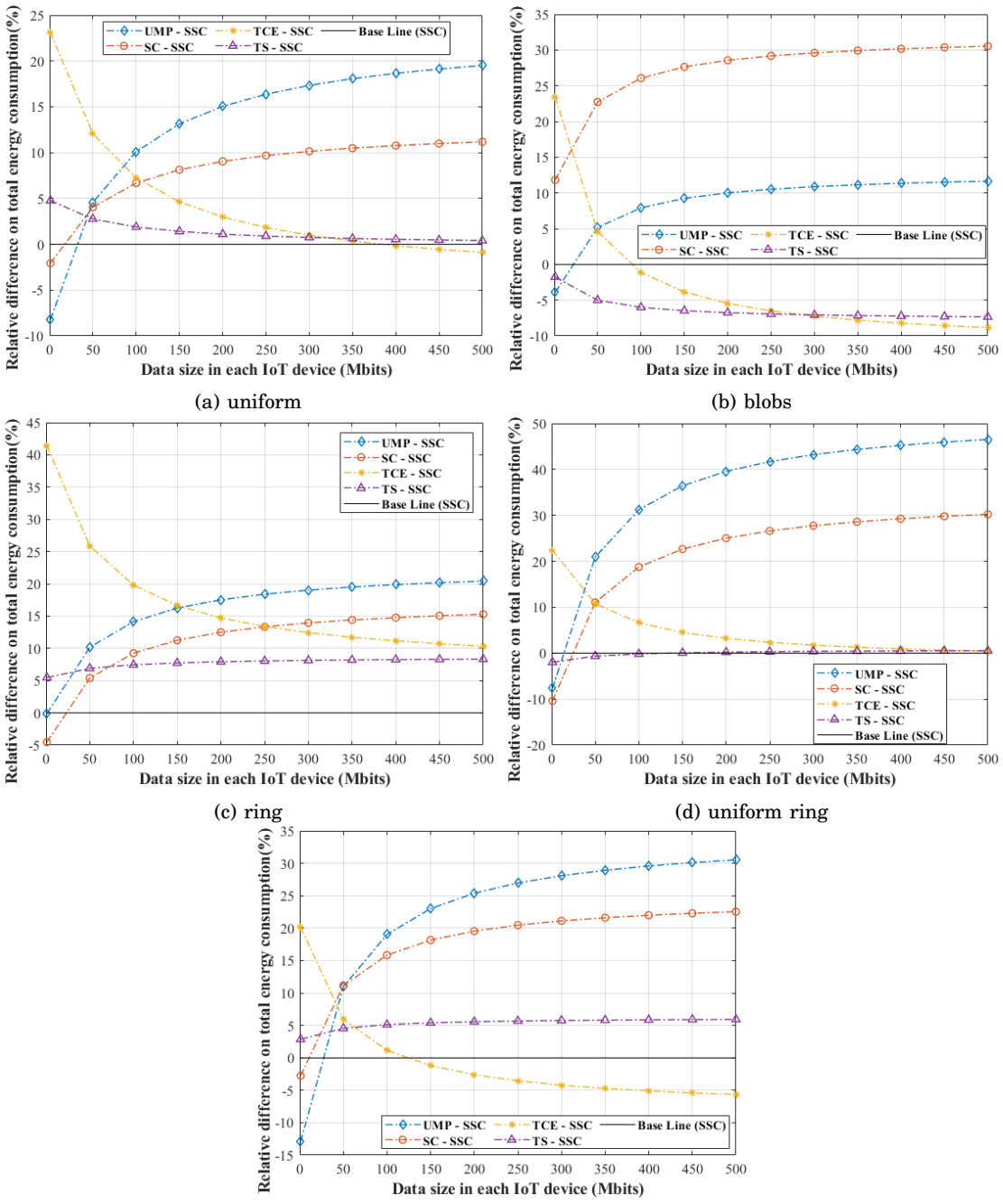
This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (Grant No.: 2020R1I1A3072688).

References

[1] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. S. Othman, A. Khreishah, and M. Guizani, "Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges," *IEEE Access*, vol. 7, pp. 48 572–48 634, 2019.

[2] L. Gupta, R. Jain, and G. Vaszkun, "Survey of important issues in uav communication networks," *IEEE Communications Surveys Tutorials*, vol. 18, no. 2, pp. 1123–1152, 2016.

[3] Y. Zeng, R. Zhang, and T. J. Lim, "Wireless communications with unmanned aerial vehicles: opportunities and challenges," *IEEE Communications Magazine*, vol. 54, no. 5, pp. 36–42, 2016.



(a) uniform

(b) blobs

(c) ring

(d) uniform ring

(e) corner

Fig. 9: Related difference in total energy consumption for different topology scenarios with the varying size of data on each IoT device

[4] S. Li *et al.*, "Dynamic online trajectory planning for a uav-enabled data collection system," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 12, pp. 13 332–13 343, 2022.

[5] X. Bresson and T. Laurent, "The transformer network for the traveling salesman problem," 2021.

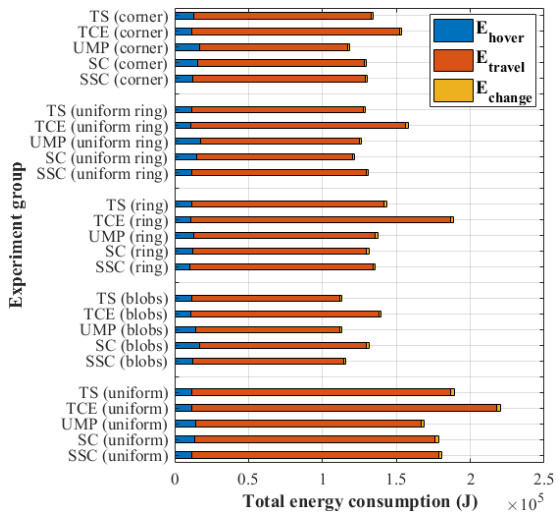
[6] M. Chen, W. Liang, and Y. Li, "Data collection maximization for uav-enabled wireless sensor networks," in *2020 29th International Conference on Computer Communications and Networks (ICCCN)*, 2020, pp. 1–9.

[7] X. Ji, X. Meng, A. Wang, Q. Hua, F. Wang, R. Chen, J. Zhang, and D. Fang, "E2pp: An energy-efficient path planning method for uav-assisted data collection," *Security and Communication Networks*, vol. 2020, p. 8850505, Dec 2020. [Online]. Available: https://doi.org/10.1155/2020/8850505

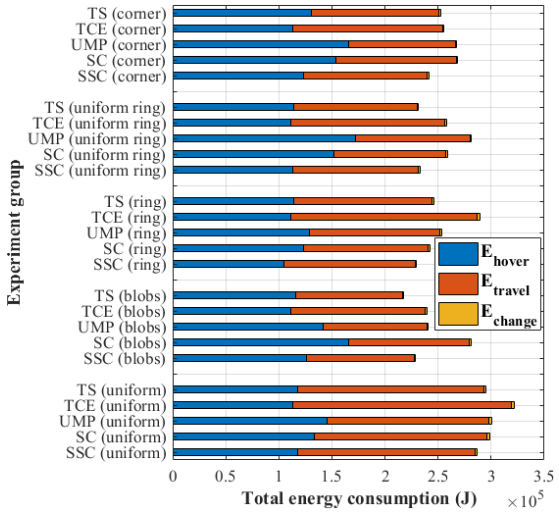
[8] J. R. Martinez-de Dios, K. Lferd, A. de San Bernabé, G. Núñez, A. Torres-González, and A. Ollero, "Cooperation between uas and wireless sensor networks for efficient data collection in large environments," *Journal of Intelligent & Robotic Systems*, vol. 70, no. 1, pp. 491–508, Apr 2013. [Online]. Available: https://doi.org/10.1007/s10846-012-9733-2

[9] D.-T. Ho, E. I. Grøtli, P. B. Sujit, T. A. Johansen, and J. B. Sousa, "Optimization of wireless sensor network and uav data acquisition," *Journal of Intelligent & Robotic Systems*, vol. 78, no. 1, pp. 159–179, Apr 2015. [Online]. Available: https://doi.org/10.1007/s10846-015-0175-5

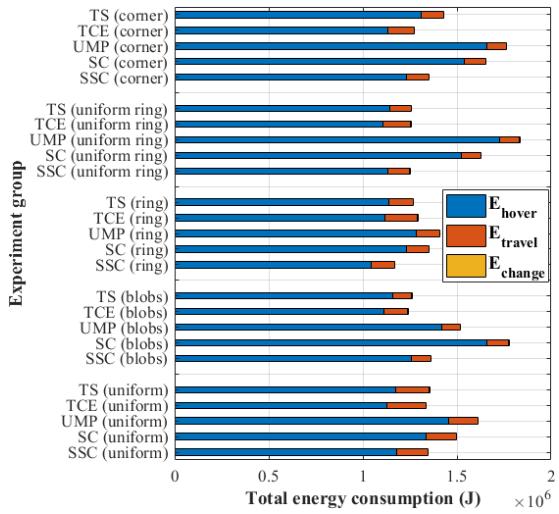
[10] M. Chen, W. Liang, and S. K. Das, "Data collection utility maximization in wireless sensor networks via efficient determination of uav hovering locations," in *2021 IEEE International Conference*



(a) 5 Mbits



(b) 50 Mbits



(c) 500 Mbits

Fig. 10: Total energy consumption situation for three different data packet configurations

on Pervasive Computing and Communications (PerCom), 2021, pp. 1–10.

[11] L. Perron *et al.*, "Or-tools," Google. [Online]. Available: https://developers.google.com/optimization/

[12] B. Zhu *et al.*, "Uav trajectory planning for aoi-minimal data collection in uav-aided iot networks by transformer," *IEEE Transactions on Wireless Communications*, vol. 22, no. 2, pp. 1343–1358, 2023.

[13] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Networks*, vol. 3, no. 3, pp. 325–349, 2005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1570870503000738

[14] H. J. Gulczynski D.J. and P. C.C., "The close enough traveling salesman problem: A discussion of several heuristics," in *[Perspectives in Operations Research: Papers in Honor of Saul Gass' 80th Birthday]*, vol. 36. Springer, 2006.

[15] M. Rezapour and D. der Naturwissenschaften, "Network design with facility location approximation and exact techniques," 2015.

[16] T. Zhang, R. Ramakrishnan, and M. Livny, "Birch: An efficient data clustering method for very large databases," in *Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data*, ser. SIGMOD '96. New York, NY, USA: Association for Computing Machinery, 1996, p. 103–114. [Online]. Available: https://doi.org/10.1145/233269.233324

[17] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96. AAAI Press, 1996, p. 226–231.

[18] "Dji website (2020), phantom 4 specifications." [Online]. Available: https://www.dji.com/phantom-4-pro-v2/specs

[19] J. Zhang, J. F. Campbell, D. C. Sweeney II, and A. C. Hupman, "Energy consumption models for delivery drones: A comparison and assessment," *Transportation Research Part D: Transport and Environment*, vol. 90, p. 102668, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361920920308531

[20] H. Mei *et al.*, "Energy-efficient segment clustering algorithm for uav trajectory," in *2022 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2022, pp. 1071–1076.

[21] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, "Mobile internet of things: Can uavs provide an energy-efficient mobile architecture?" in *2016 IEEE Global Communications Conference (GLOBECOM)*, 2016, pp. 1–6.

[22] A. Merwaday and I. Guvenc, "Uav assisted heterogeneous networks for public safety communications," in *2015 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2015, pp. 329–334.