2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

UBAT: On Jointly Optimizing UAV Trajectories and Placement of Battery Swap Stations

Myounggyu Won
Department of Computer Science, University of Memphis, Memphis, TN, United States
mwon@memphis.edu

Abstract— **Unmanned aerial vehicles (UAVs) have been widely used in many applications. The limited flight time of UAVs, however, still remains as a major challenge. Although numerous approaches have been developed to recharge the battery of UAVs effectively, little is known about optimal methodologies to deploy charging stations. In this paper, we address the charging station deployment problem with an aim to find the optimal number and locations of charging stations such that the system performance is maximized. We show that the problem is NP-Hard and propose UBAT, a heuristic framework based on the ant colony optimization (ACO) to solve the problem. Additionally, a suite of algorithms are designed to enhance the execution time and the quality of the solutions for UBAT. Through extensive simulations, we demonstrate that UBAT effectively performs multi-objective optimization of generation of UAV trajectories and placement of charging stations that are within 8.3% and 7.3% of the true optimal solutions, respectively.**

I. INTRODUCTION

With recent breakthroughs in design and production of UAVs, UAVs are increasingly used in many applications such as military surveillance [1], disaster response [2], oil gas pipe inspection [3], precision agriculture [4], and delivery of goods [5]. The Federal Aviation Administration (FAA) estimates that commercial UAVs will grow 10-fold between 2016 and 2021, and the global market revenue of UAVs will rise to \$11.2 billion by 2020 [6]. One of the major obstacles that constrains the huge potential of UAVs is the limited flight time. This is a significant problem as more and more UAV applications require coverage of a large geographical area, lengthening the mission duration well beyond the battery capacity of UAVs [7].

Numerous solutions have been developed to extend the flight time of UAVs focusing on utilization of charging stations that recharge/replace the battery of UAVs [8][9][10]. While these solutions have successfully increased the operation time of UAVs, there is still a significant knowledge gap on how to deploy charging stations to maximize the system performance. Specifically, existing approaches are based on a simplifying assumption for deploying charging stations. Wei and Isler propose a solution based on a single charging station [11]. Some solutions deploy charging stations only at fixed locations [12] such as at the center of a field [13] and the base station [14]. Yu *et al.* develop a solution based on unmanned ground vehicle (UGV)-based charging, but these UGVs are allowed to visit only the sites to be visited by the UAVs [15]. It remains as a challenge to develop a solution that finds the optimal UAV trajectories with no constraints on the number and locations of charging stations.

In this paper, we address the problem of deploying charging stations that maximizes the system performance specifically concentrating on generating optimal UAV trajectories, while minimizing the number of deployed charging stations. More specifically, we formulate the problem of jointly optimizing UAV Trajectories and Locations of Battery swap Stations (**TLBS**) as an optimization problem, and develop an optimization framework to solve the problem. Especially, since finding a shortest trajectory for a single UAV visiting each region of interest (ROI) and returning to the base station, without even considering charging stations, is essentially the travelling salesperson problem (TSP) which is NP-Hard, ***we propose a heuristic solution called UBAT based on Ant Colony Optimization (ACO) motivated by the fact that ACO is well known to solve TSP very effectively [16].*** The technical contributions of this paper are that the standard ACO is adapted to account for unique requirements for solving the **TLBS** problem especially in calculating the probabilities for path selection and updating the pheromone trails: (1) constrained energy of UAVs, (2) simultaneous coverage of ROIs by multiple UAVs, and (3) joint optimization of UAV trajectories and number of charging stations.

The proposed framework is fine-tuned with novel algorithms that are designed to enhance the convergence speed and the quality of solutions. More specifically, an algorithm is developed to minimize the number of candidate locations for deploying charging stations, thereby reducing the search space and lowering the computational overhead. We also design an algorithm that effectively tunes the parameters for the proposed framework to enhance the quality of solutions. Additionally, the 2-OPT local search [17] is effectively incorporated to further improve the quality of solutions. Extensive simulations are conducted to evaluate the performance of UBAT as well as the supporting algorithms. The results demonstrate that the proposed approach effectively performs joint optimization of UAV trajectories and placement of charging stations that are within 8.3% and 7.3% of the true optimal solutions, respectively.

We define the **TLBS** problem in Section II and present the details of UBAT in Section III followed by descriptions of supporting algorithms in Section IV. Simulation results are presented in Section V. We then conclude in Section VI.

II. TLBS Problem

A. System Model

Consider a target area A with an arbitrary shape in $\mathbb{R}^2$. The target area is divided into a two-dimensional grid of square cells, *i.e.,* the target area is represented as a set $A = \{a_1, a_2, ..., a_{NS}\}$ where each element is a subarea (cell), and NS is the number of subareas. Among the cells are regions of interests (ROIs) denoted by a set $R = \{r_1, r_2, ..., r_{NR}\} \subseteq A$, where NR is the number of ROIs. We introduce the function $\mathcal{F}_{dist} : A, A \to \mathbb{R}$ which defines the Euclidean distance between two cells. More specifically, $\mathcal{F}_{dist}(a_i, a_j)$ means the Euclidean distance between the centers of two cells a_i and a_j.

There are NU UAVs denoted by a set $U = \{u_1, u_2, ..., u_{NU}\}$. Assume a typical UAV application where UAVs fly over a target area, perform sensing, and transmit sensed data to the base station. We assume that UAVs fly H meters high from the ground. As such, the sensing coverage COV is $\pi \cdot (H \cdot \tan(\frac{\theta}{2}))^2$ where θ is the angle of the sensing cone. The sensing coverage is sufficiently large to cover a whole ROI when a UAV is above the center of the ROI. Fig. 1 illustrates a target area divided into cells, three ROIs in the target area, and the sensing coverage. The time is discretized into time slots $T = \{t_1, t_2, ...\}$ where the length of a time slot is denoted by LT. In particular, let t_{finish} be the time slot when all ROIs have been fully covered.

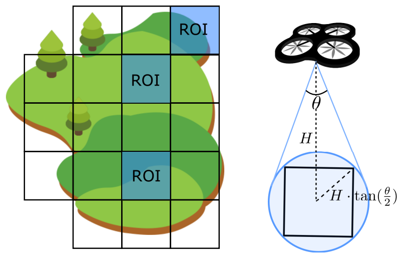

Fig. 1. An illustration of a target area, ROIs, and sensing coverage.

The flight time of a UAV is limited due to its energy capacity. To support continuous operation of UAVs, NC charging stations are deployed in the target area which are denoted by a set $C = \{c_1, c_2, ..., c_{NC}\}$. More precisely, we assume that these charging stations automatically replace the battery of a UAV when it is landed on the charging station.

Let E_{MAX} be the maximum energy capacity of a UAV. Also, let E_{u_i,t_j} be the remaining energy of a UAV $u_i \in U$ at time slot $t_j \in T$. Assume that all UAVs are fully charged before operation, *i.e.,* $E_{u_i,t_0} = E_{MAX}, \forall u_i \in U$. We define γ_{fly} be the energy loss rate while a UAV is flying, which can be updated based on the factors that affect energy consumption such as wind. Each UAV u_i is in one of these states $S = \{s_{fly}, s_{rec}\}$ at each time slot. The state s_{fly} indicates that a UAV is flying, and the state s_{rec} indicates that a UAV is landed on a charging station to get its battery replaced.

A function $\mathcal{F}_{fly} : U, A, T \to \{0, 1\}$ is defined to specify whether a UAV is flying over a certain cell at a certain time slot. For example, $\mathcal{F}_{fly}(u_i, a_j, t_k) = 1$ if UAV u_i is flying over cell a_j at time slot t_k, and $\mathcal{F}_{fly}(u_i, a_j, t_k) = 0$ if it is landed on a charging station located in a cell a_j. The residual energy of a UAV u_i at time slot t_j, *i.e.,* E_{u_i,t_j} can then be defined as follows.

$$\begin{aligned} E_{u_i,t_k} &= E_{u_i,t_{k-1}} - \gamma_{fly} \cdot \mathcal{F}_{fly}(u_i, a_j, t_k) \\ &\quad + (E_{MAX} - E_{u_i,t_{k-1}}) \cdot (1 - \mathcal{F}_{fly}(u_i, a_j, t_k)). \end{aligned} \tag{1}$$

Here, $E_{u_i,t_{j-1}}$ is the residual energy at the previous time slot. $\gamma_{fly} \cdot f_{fly}(u_i, t_j)$ represents the energy loss for flying during the time slot, and $(E_{MAX} - E_{u_i,t_{j-1}}) \cdot (1 - f_{fly}(u_i, t_j))$ indicates replacement of the battery during the time slot.

We introduce the function $\mathcal{F}_{uloc} : U, A, T \to \{0, 1\}$ indicating whether the UAV is located in a certain cell at a certain time slot. For example, $\mathcal{F}_{uloc}(u_i, a_j, t_k) = 1$ means that UAV $u_i \in A$ is in a cell $a_j \in A$ at time slot $t_k \in T$, and otherwise $\mathcal{F}_{uloc}(u_i, a_j, t_k) = 0$. We also define the function $\mathcal{F}_{cloc} : A \to \{0, 1\}$ indicating that a charging station is deployed in a certain cell. More precisely, $\mathcal{F}_{cloc}(a_i) = 1$ if a charging station is in the cell $a_i \in A$, otherwise $\mathcal{F}_{cloc}(a_i) = 0$. Let us define another function $\mathcal{F}_{mov} : U, T \to \mathbb{R}$ which indicates the Euclidian distance moved during a time slot. For example $\mathcal{F}_{mov}(u_i, t_j)$ means the distance moved by UAV $u_i \in U$ during time slot $t_j \in T$. Let v_{MAX} be the maximum speed of the UAV.

B. Problem Formulation

Having defined all notations and functions, the problem of jointly optimizing UAV Trajectories and Locations of Battery swap Stations (**TLBS**) is formulated as follows.

$$\text{minimize} \quad \{t_{finish}, \sum_{i=1}^{NS} \mathcal{F}_{cloc}(a_i)\} \tag{2}$$

$$\sum_{j=1}^{NS} \mathcal{F}_{uloc}(u_i, a_j, t_k) \leq 1, \forall u_i \in U, \forall t_k \in T \tag{3}$$

$$\sum_{i=1}^{NU} \sum_{k=1}^{finish} \mathcal{F}_{uloc}(u_i, a_j, t_k) \geq 1, \forall a_j \in R \tag{4}$$

$$0 < E_{u_i,t_j} \leq E_{MAX}, \forall u_i \in U, \forall t_j \in T \tag{5}$$

$$\mathcal{F}_{mov}(u_i, t_j) \leq LT \cdot v_{MAX}, \forall u_i \in U, \forall t_j \in T \tag{6}$$

$$\begin{aligned} \mathcal{F}_{cloc}(a_j) + \mathcal{F}_{fly}(u_i, a_j, t_k) \geq 1, \\ \forall u_i \in U, \forall a_j \in A, \forall t_k \in T \end{aligned} \tag{7}$$

$$\text{Equation 1} \tag{8}$$

Constraint 3 asserts that each UAV can be located in only one cell at each time slot. Constraint 4 means that each ROI is covered by at least one UAV when the UAV operation is finished. Constraint 5 states that each UAV does not deplete its energy completely during operation, and the residual energy should not be greater than the maximum energy capacity. Constraint 6 assures that each UAV does not move faster than the maximum speed v_{MAX}. Constraint 7 means that if a UAV is not flying, *i.e.,* landed in a cell for recharging, there should be a charging station in that cell. Finally, Constraint 8 specifies how the residual energy of each UAV is updated at each time slot. The **TLBS** problem is NP-Hard because it contains, as a simplified instance, the traveling salesman problem (TSP) [18]. As such, we develop a heuristic framework based on the ant colony optimization (ACO) motivated by the fact that ACO is particularly effective in solving the TSP problem [16].

III. Proposed Approach

A. Path Selection

We represent UAVs and ROIs as ants and food sources, respectively. For simplicity, we assume that UAVs fly at a speed of v_{MAX} (Constraint 6), which can be easily replaced by a vector to represent the time-varying speed. We follow the energy model defined in Eq. 1 to respect Constraint 8. Compared to ACO with ants, we leverage two advantages of UAVs. First, UAVs have memory; they can keep track of visited ROIs so that they will visit only ROIs that have not been covered by any other UAV. Second, UAVs have computational capabilities; they can compute the distance between two ROIs. Based on these advantages, the probability that a UAV in cell a a_i selects a ROI a_j to visit is defined as follows.

$$p_{ij} = \begin{cases} \frac{\tau_{ij}^{\alpha} \cdot \eta_{ij}^{\beta}}{\sum_{k \in R_{visit}} \tau_{ik}^{\alpha} \cdot \eta_{ik}^{\beta}} & \text{if } a_j \in R_{visit} \\ 0 & \text{otherwise} \end{cases} \quad (9)$$

Here R_{visit} is the set of ROIs that have not been visited by any UAV (Constraint 4). τ_{ij} is the pheromone intensity of an edge connecting two cells a_i and a_j. η_{ij} is the inverse of the Euclidean distance between two cells a_i and a_j namely $\frac{1}{\mathcal{F}_{dist}(a_i,a_j)}$, which is used to ensure that a shorter edge is selected in finding a path. Two parameters α and β are introduced to adjust the influence of the pheromone and the distance between two cells, respectively. After calculating the probabilities for all ROIs, a UAV selects a ROI with the highest probability and moves to the selected ROI.

If a UAV does not have enough energy to reach the selected ROI, the state of the UAV is changed to the charging mode, and the UAV finds a set of cells that it can reach with its residual energy (Constraint 5). A cell is selected from those candidate cells using a slightly modified probability model compared to Eq. 9, and then the UAV moves to the selected cell in which a charging station is placed (Constraint 7). Specifically for the modified probability model, η_{ij} is set to the Euclidean distance between two ROIs (i.e., $\eta_{ij} = \mathcal{F}_{dist}(a_i, a_j)$), instead of the inverse of it as specified in Eq. 9, to minimize the number of deployed charging stations by assuring that a longer edge is preferred in the charging mode, *i.e.,* we allow UAVs to fly as far as possible with the given residual energy because the battery will be replaced anyways.

B. Trail Update

The pheromone trails are updated in two ways: evaporation and reinforcement. The former mimics the natural evaporation of pheromone. More specifically, pheromone τ_{ij} between two cells a_i and a_j are reduced if $\tau_{ij} > 0$ based on the formula: $\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \rho \cdot \tau_0$, where ρ is a parameter that controls the speed of pheromone reduction, and τ_0 is the initial pheromone. Evaporation is particularly useful for preventing one path from dominating other possible solutions. Due to evaporation, other paths are explored, potentially leading to a better solution.

Once a path P is discovered, pheromone trails on that path are strengthened, which is called the reinforcement. The reinforcement process is used to encourage the use of shorter paths and to increase the chance of using the edges of the currently known shortest path in finding potentially better paths subsequently. Formally, the pheromone trail of each edge of the selected path is reinforced as follows: $\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \rho \cdot \frac{Q}{L}$, where L is the length of path P; Q is a system parameter – Details on determining an appropriate value for the parameter is discussed in Section V-B.

C. Multiple UAVs with Minimum Charging Stations

We now extend the algorithm to account for general scenarios where multiple UAVs collaborate to cover ROIs with an objective of minimizing the number of deployed charging stations. More specifically, each UAV u_i visits a portion of ROIs and ends up with having an individual path denoted by P_{u_i}. Denote the length of path P_{u_i} for UAV $u_i \in U$ be L_{u_i}. Then, to minimize t_{finish}, we try to minimize the maximum path length of all UAVs, *i.e.,* $\max_{1 \leq i \leq NU} L_{u_i}$, rather than minimizing L. Consequently, the pheromone updating formula is rewritten as follows: $\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \rho \cdot \frac{Q}{\max_{1 \leq i \leq NU} L_{u_i}}$.

We then take into account the number of deployed charging stations in the pheromone updating rule for reinforcement. If we do not constrain the number of charging stations in finding a path, it may increase arbitrarily. To this end, UBAT tries to make as many of the charging stations be shared by multiple UAVs such that the path length is also optimized. The idea to implement this is to include the number of deployed charging stations NC in the pheromone updating formula. More specifically, we aim to find cells to deploy charging stations such that both t_{finish} and the number of deployed charging stations NC are minimized, as specified in the objective function (Eq. 2). In order to encourage the use of a path that minimizes both t_{finish} and NC, we can rewrite the pheromone updating formula as follows: $\tau_{ij} = (1-\rho) \cdot \tau_{ij} + \rho \cdot (\frac{Q_1}{\max_{1 \leq i \leq NU} L_{u_i}} + \frac{Q_2}{NC})$.

Algorithm 1 summarizes the operation of UBAT. Starting from the current cell $c \in A$, each UAV $u \in U$ selects the next cell $r \in R$ to visit using Eq. 9 (Line 6). If the selected cell is reachable with the remaining energy E_u, the edge connecting c and r, namely Seg_{cr} is added to the path P_u of UAV u (Lines 7-9). On the other hand, if r is not reachable, the

Algorithm 1: ACO for TLBS problem

```
1  begin
2      Initialize $\alpha$, $\beta$, $\tau_0$, $Q_1$, $Q_2$ and $\rho$.
3      while $time < MAX\_ITERATION$ do
4          while $R_{visit} \neq \emptyset$ do
5              for each $u \in U$ do
6                  Find a cell $r \in R$ to visit using Eq. 9.
7                  if $r$ is reachable with $E_u - E_{threshold}$
                     then
8                      $R_{visit} \leftarrow R_{visit} - \{r\}$.
9                      $c \leftarrow$ current cell;
                       $P_u \leftarrow P_u \cup Seg_{cr}$.
10                 else
11                     Find a set of cells $\hat{A} \subseteq A$ that are
                         reachable with residual energy.
12                     Locate a cell $a \in \hat{A}$ to place a
                         charging station using the
                         modified version of Eq. 9.
13                     $c \leftarrow$ current cell;
                       $P_u \leftarrow P_u \cup Seg_{ca}$.
14                     $NC \leftarrow NC + 1$.
15         Update $\tau_{ij}$ based on evaporation $\forall i, j \in A$ if
             $Seg_{ij} \notin \bigcup_{u_i \in U} P_{u_i}$.
16         Update $\tau_{ij}$ based on reinforcement $\forall i, j \in A$
             if $Seg_{ij} \in \bigcup_{u_i \in U} P_{u_i}$.
17         $P_u \leftarrow \emptyset, \forall u \in U$.
```

algorithm locates a cell a to place a charging station based on the modified version of the probability model Eq. 9 (Lines 11-14). More precisely, a charging station is selected such that both NC and $\max_{1 \leq i \leq NU} L_{u_i}$ are minimized, and the path segment connecting cells c and a becomes part of P_u. Since a new charging station has been placed in cell a, the number of charging stations is incremented by one. This process is repeated until all ROIs are visited by UAVs, and as a result, we obtain the trajectories for all UAVs and the cells that have charging stations. Given the discovered paths $P_u, \forall u \in U$, the pheromone trails are updated based on evaporation and reinforcement (Lines 15-16). This whole process is repeated until we reach $MAX\_ITERATION$ (Line 3).

IV. Enhancement Techniques

Reducing Search Space: We develop an algorithm to improve the execution time of UBAT by reducing the overhead for finding a set of candidate cells $\hat{A}$ for deploying a charging station (Lines 10-14 of Algorithm 1). The key idea is to make the search space smaller based on the observation that a charging station can be deployed only within the convex hull of ROIs. Consequently, only the cells within the convex hull are considered in finding $\hat{A}$.

We can easily prove that deploying charging stations only inside the convex hull of ROIs does not influence the quality of solution. Assume in contradiction that a charging station is placed outside the convex hull, *e.g.,* c_1 in Fig. 2 to allow the

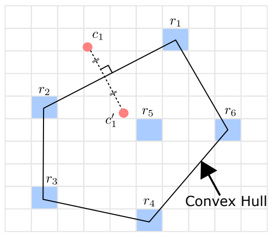

Fig. 2. An illustration of candidate cell selection.

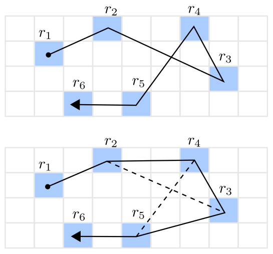

Fig. 3. An illustration of the 2-OPT local search.

UAV to move from r_1 to r_2. We can always find c_1' such that $\mathcal{F}_{dist}(r_1, c_1) + \mathcal{F}_{dist}(c_1, r_2) = \mathcal{F}_{dist}(r_1, c_1') + \mathcal{F}_{dist}(c_1', r_2)$. Furthermore, c_1' is a better choice because it is closer to other ROIs considering that it may be used by other UAVs.

Tuning Parameters: Another key factor that affects the performance of UBAT is the parameters Q_1 and Q_2. If we choose too large values for Q_1 and Q_2, *i.e.,* $\tau_0 << \frac{Q_1}{\max_{1 \leq i \leq NU} L_{u_i}} + \frac{Q_2}{NC}$, the pheromone intensity of a new path would become too strong. As a result, it will take long for the pheromone of a newly found path to evaporate down to the level that other potential paths can be explored. On the other hand, if the values for the parameters are too small, *i.e.,* $\tau_0 >> \frac{Q_1}{\max_{1 \leq i \leq NU} L_{u_i}} + \frac{Q_2}{NC}$, the pheromone intensity of a new path would become too small leading to the situation where a potentially good solution is dropped too early, and as such, UBAT misses a chance to validate that it was actually a good solution.

We design an iterative approach to find the values for Q_1 and Q_2 such that $\tau_0 \approx \frac{Q_1}{\max_{1 \leq i \leq NU} L_{u_i}} + \frac{Q_2}{NC}$. More specifically, we set τ_0 to 1 and start UBAT with arbitrary values for Q_1 and Q_2. After only a few iterations, we get intermediate solutions a and b for $\max_{1 \leq i \leq NU} L_{u_i}$ and NC, respectively. We use these intermediate solutions as new values for the parameters (*i.e.,* $Q_1 \leftarrow a, Q_2 \leftarrow b$), and then, we repeat the same process to obtain new values for the parameters. Through experiments in Section V-B, we validate that this approach is quick and effective requiring only one or two iterations for finding adequate values of the parameters.

Improving paths: To make UBAT run faster and obtain a higher quality solution, we perform correction on a path found in each iteration by applying the 2-OPT local search algorithm [17]. The algorithm was first proposed by Croes for improving the quality of a solution for TSP [19]. The idea is to detect a path that crosses over itself and perform reordering of cells so that the cross over is removed. More precisely, two edges that cross each other are removed from a path, and then the resulting two path segments after removal of the edges are reconnected such that the cross over no longer exists. For example in Fig. 3, an edge $r_2 \rightarrow r_3$ is replaced by $r_2 \rightarrow r_4$; and an edge $r_4 \rightarrow r_5$ is replaced by $r_3 \rightarrow r_5$, resulting in a shorter path with no cross over. Unlike the traditional 2-OPT local search, however, we ensure that the output of the algorithm does not result in increased path length for other UAVs and deployment of additional charging stations.

V. Evaluation

We implemented UBAT using C++ on a PC equipped with a 2.5GHz dual-core Intel processor, and 16GB RAM. A field with dimensions of 20×20km^2 was created which was divided into cells with dimensions of 1×1km^2, *i.e.,* 400 cells in the field. Random scenarios were created by selecting 10 ROIs randomly. UAVs started operation from one of the randomly selected ROIs. We conducted simulations with different numbers of UAVs for which the batteries were fully charged. The maximum distance that a fully charged UAV can fly was set to 5km. The sensing range of a UAV was configured to fully cover a cell when it hovers over the cell. The parameters for ACO were set as $\alpha = 2, \beta = 2$, and $\rho = 0.3$. The max iteration count was set to 30,000. The metrics measured were (1) the length (km) of the longest path of all UAVs, *i.e.,* (which will be referred to as 'path length' hereafter), and (2) the number of deployed charging stations (CS). The path length and number of CS of UBAT were compared with the true optimal solutions. The effectiveness of the proposed enhancement techniques was also evaluated.

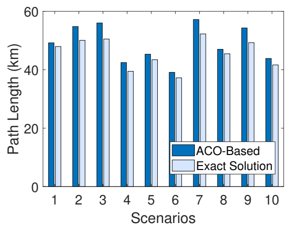

Fig. 4. Path length for 1 UAV.

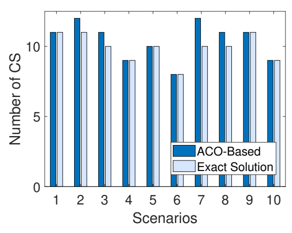

Fig. 5. Number of CS for 1 UAV.

A. Comparison with Exact Algorithm

The performance of UBAT was compared with the exact algorithm which finds the solution using the brute force search. A challenge for running the exact algorithm is the huge computation time. Fortunately, the true optimal solutions for a single-UAV scenario can be easily and quickly calculated [20]. Specifically, the optimal path of a single UAV is simply the output of TSP with a set of ROIs as input, reducing the search space from 400 (number of cells) to 10 (number of ROIs). Calculating path lengths for all possible permutations for 10 ROIs completes in a few seconds. Given the shortest path, the minimum number of CS can be computed in linear time by deploying a CS at a point on the shortest path that is farthest from the current position of a fully charged UAV [20].

Figs. 4 and 5 show the path length and number of CS for UBAT and the exact algorithm. On average, the path length of UBAT was 6.3% greater with STDEV of 2.6% compared with the true optimal solutions, and the number of CS was 4.3% greater with STDEV of 6.0%. Putting in different terms, UBAT had a 3.1km longer path and required 0.5 additional charging stations on average in comparison with the true optimal solutions under single-UAV scenarios.

We then measured the path length and the number of CS for multi-UAV scenarios. Unlike single-UAV scenarios, however, we cannot simply apply the TSP algorithm to find the true optimal solution for multi-UAV scenarios resulting in huge computation time. A quick, yet effective solution to this challenge is to test the algorithms with 'semi' random scenarios. Specifically, 5 ROIs are randomly selected from the top of the field, and the other 5 ROIs are randomly selected from the bottom of the field such that these two sets of ROIs are individually covered by each UAV. This way the 2-UAV scenarios can be divided into two separate single-UAV scenarios that allows us to apply the TSP algorithm to obtain the true optimal solutions quickly, while UBAT is not aware of this configuration and running normally. A similar simulation setting was used for 4-UAV scenarios, *i.e.,* we deployed 10 ROIs randomly in NE, NW, SE, SW regions of the field such that ROIs in each region can be independently covered by each UAV. We leave the work of obtaining the true optimal solutions for more complicated scenarios with a large number of UAVs using a high performance computing system as our future work.

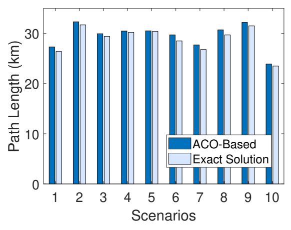

Fig. 6. Path length for 2 UAVs.

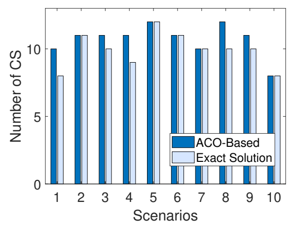

Fig. 7. Number of CS for 2 UAVs.

Figs. 6 and 7 show the impressive performance of UBAT for 2-UAV scenarios. The average path length of UBAT was 2.2% greater with STDEV of 1.1% in comparison with the exact algorithm. The average number of CS was 7.3% greater with STDEV of 8.4% compared with the exact algorithm. The larger percentage difference for the number of CS compared with the results for the single-UAV scenarios can be attributed to the fact that fewer charging stations were deployed for the 2-UAV scenarios because only a half of the field was covered by a UAV. Also, even though the percentage difference for the number of CS may seem much greater than that for the path length, it means only 0.8 additional charging stations on average in comparison with the true optimal solutions.

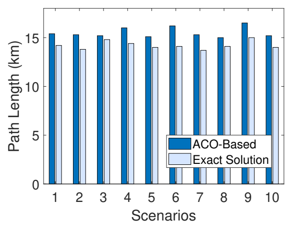

Fig. 8. Path length for 4 UAVs.

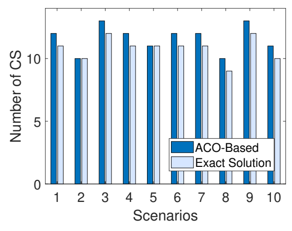

Fig. 9. Number of CS for 4 UAVs.

Similarly, Figs. 8 and 9 show the results for the 4-UAV scenarios. The average path length and the number of CS for the proposed framework were 8.3% (STDEV 2.8%) and 6.7% (STDEV of 3.6%) greater in comparison with the true optimal solutions, respectively. In other words, the average path length of UBAT was 1.3km longer with 0.8 additional charging stations compared with the true optimal solutions. Overall, the results demonstrate that UBAT produces very high-quality solutions with slightly longer path length and less than 1 more charging stations compared with the true optimal solutions in our simulation environments.

B. *Effect of Enhancement Techniques*

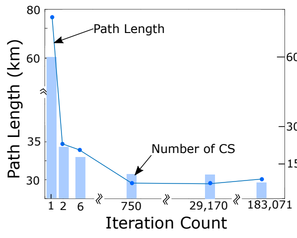

Fig. 10. Effect of iteration count.

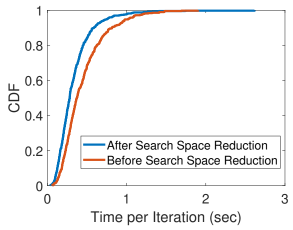

Fig. 11. CDF of execution time.

Effect of Max Iterations: We investigate the effect of the maximum iteration count (the default is set to 30,000). It is an interesting analysis because the longer we run UBAT, the solution keeps improving. Specifically, we increased the max iteration count to 400,000 while fixing other settings for a 2-UAV scenario. We observed the following from the results (Fig. 10): (1) a sufficiently good solution is obtained within the first few iterations; (2) after that, the solution keeps improving but very slightly. In this particular example, the path length was significantly decreased in the first 750 iterations, and it was reduced only by 0.9% between 750th and 68,432th iterations (and virtually no improvement after that); similarly the number of CS stayed the same after the first few iterations, although in some cases, we observed that the number of CS was decreased after a very large number of iterations at the cost of the increased path length. Overall, while the solution keeps improving as we increase the maximum iteration count, the advantage is not obvious considering that major progress is made in the first few iterations and using a large maximum iteration count significantly increases the execution time.

Effect of Search Space Reduction: To understand the effect of the search space reduction algorithm, we measured the execution time (*i.e.,* the time it takes to run a single iteration for UBAT) with and without applying the algorithm under the 2-UAV scenarios. The cumulative distribution function (CDF) of the execution time is shown in Fig. 11. The results show that the average time for running a single iteration without the algorithm was 0.44 seconds, and it was decreased to 0.33 seconds when the search space reduction algorithm was used, indicating 25% faster execution time.

Effect of Parameter Tuning: We evaluate the performance of the parameter tuning algorithm. We started with some arbitrary values for the parameters Q_1 and Q_2. We then executed UBAT only for a few iterations, *i.e.*, 1,000

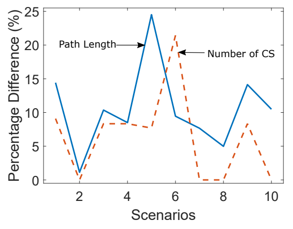

Fig. 12. Effect of parameter tuning.

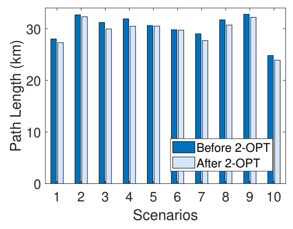

Fig. 13. Effect of path correction.

iterations. As observed in the experiments for the max iteration count, a sufficiently good solution is obtained in the first few iterations. The obtained intermediate solution, *i.e.,* the path length, and the number of CS were used as new values for parameters Q_1, and Q_2, respectively. We then ran UBAT with the new parameters and compared the results with those obtained with arbitrarily selected parameters. The results in Fig. 12 show that the path length and the number of CS were improved by up to 24.5% and 21.4% (10.5% and 6.3% on average), respectively, when the parameter tuning algorithm was applied.

We then evaluate the performance gain when the parameter tuning algorithm is applied repeatedly. An interesting observation was that the performance gain was very marginal even from the second time running the parameter tuning algorithm. Specifically, the path length and the number of CS were improved only by up to 0.3% on average. In the third time for tuning the parameters, virtually no improvement was observed, indicating that running the parameter tuning process once or twice suffices to obtain good parameters. Note that all simulation results were obtained using this parameter tuning method, showing the effectiveness and practicality of the proposed approach.

Effect of Path Correction: We analyze the effect of the path improvement algorithm. We measured the path length with and without applying the algorithm, while fixing all other settings for the 2-UAV scenarios. It should be noted that the number of CS was not measured because the path improvement algorithm does not increase the number of CS at the cost of improving the path. The results are depicted in Fig. 13. The path length was decreased by 2.5% on average with STDEV of 1.6% when the path improvement algorithm was applied. The results indicate that by skipping low-quality solutions (*e.g.,* paths with self-crossings), the search space is explored more effectively, leading to better solutions.

VI. Conclusion

We have presented UBAT, an optimization framework that jointly optimizes UAV trajectories and locations of charging stations. Enhancement techniques are developed to improve the convergence time and quality of solutions. Through extensive simulations, we demonstrate that UBAT effectively calculates the UAV trajectories and placement of charging stations. The future work is to perform theoretical analysis on the algorithm complexity regarding the number of UAVs, field size, and scale of discrete grids and to conduct experiments based on actual UAV robotic platforms.

REFERENCES

[1] D. Orfanus, E. P. de Freitas, and F. Eliassen, “Self-organization as a supporting paradigm for military uav relay networks,” *IEEE Communications Letters*, vol. 20, no. 4, pp. 804–807, 2016.

[2] M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, “Help from the sky: Leveraging uavs for disaster management,” *IEEE Pervasive Computing*, no. 1, pp. 24–32, 2017.

[3] D. Hausamann, W. Zirnig, G. Schreier, and P. Strobl, “Monitoring of gas pipelines–a civil uav application,” *Aircraft Engineering and Aerospace Technology*, vol. 77, no. 5, pp. 352–360, 2005.

[4] P. Tokekar, J. Vander Hook, D. Mulla, and V. Isler, “Sensor planning for a symbiotic uav and ugv system for precision agriculture,” *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1498–1511, 2016.

[5] C. C. Murray and A. G. Chu, “The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery,” *Transportation Research Part C: Emerging Technologies*, vol. 54, pp. 86–109, 2015.

[6] “Drone trends,” https://blog.nationaldrones.com.au/drone-trends-2018, accessed: 2018-12-24.

[7] J. Gong, T.-H. Chang, C. Shen, and X. Chen, “Flight time minimization of uav for data collection over wireless sensor networks,” *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 9, pp. 1942–1954, 2018.

[8] K. A. Swieringa, C. B. Hanson, J. R. Richardson, J. D. White, Z. Hasan, E. Qian, and A. Girard, “Autonomous battery swapping system for small-scale helicopters,” in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 3335–3340.

[9] K. A. Suzuki, P. Kemper Filho, and J. R. Morrison, “Automatic battery replacement system for uavs: Analysis and design,” *Journal of Intelligent & Robotic Systems*, vol. 65, no. 1-4, pp. 563–586, 2012.

[10] M. Lu, M. Bagheri, A. P. James, and T. Phung, “Wireless charging techniques for uavs: A review, reconceptualization, and extension,” *IEEE Access*, 2018.

[11] M. Wei and V. Isler, “Coverage path planning under the energy constraint,” in *Proc. of ICRA*, 2018.

[12] C.-M. Tseng, C.-K. Chau, K. Elbassioni, and M. Khonji, “Autonomous recharging and flight mission planning for battery-operated autonomous drones,” *arXiv preprint arXiv:1703.10049*, 2017.

[13] A. Trotta, M. Di Felice, F. Montori, K. R. Chowdhury, and L. Bononi, “Joint coverage, connectivity, and charging strategies for distributed uav networks,” *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 883–900, 2018.

[14] J. Scherer and B. Rinner, “Short and full horizon motion planning for persistent multi-uav surveillance with energy and communication constraints,” in *Proc. of IROS*, 2017.

[15] K. Yu, A. K. Budhiraja, and P. Tokekar, “Algorithms for routing of unmanned aerial vehicles with mobile recharging stations,” in *Proc. of ICRA*, 2018.

[16] M. Dorigo and L. M. Gambardella, “Ant colonies for the travelling salesman problem,” *biosystems*, vol. 43, no. 2, pp. 73–81, 1997.

[17] C. Voudouris and E. Tsang, “Guided local search and its application to the traveling salesman problem,” *European journal of operational research*, vol. 113, no. 2, pp. 469–499, 1999.

[18] M. Dorigo and L. M. Gambardella, “Ant colony system: a cooperative learning approach to the traveling salesman problem,” *IEEE Transactions on evolutionary computation*, vol. 1, no. 1, pp. 53–66, 1997.

[19] G. A. Croes, “A method for solving traveling-salesman problems,” *Operations research*, vol. 6, no. 6, pp. 791–812, 1958.

[20] C.-S. Liao, S.-H. Lu, and Z.-J. M. Shen, “The electric vehicle touring problem,” *Transportation Research Part B: Methodological*, vol. 86, pp. 163–180, 2016.