Coverage of an Environment Using Energy-Constrained Unmanned Aerial Vehicles

Kevin Yu Jason M. O'Kane Pratap Tokekar

***Abstract*— We study the problem of covering an environment using an Unmanned Aerial Vehicle (UAV) with limited battery capacity. We consider a scenario where the UAV can land on an Unmanned Ground Vehicle (UGV) and recharge the onboard battery. The UGV can also recharge the UAV while transporting the UAV to the next take-off site. We present an algorithm to solve a new variant of the area coverage problem that takes into account this *symbiotic* UAV and UGV system. The input consists of a set of *boustrophedon cells* — rectangular strips whose width is equal to the field-of-view of the sensor on the UAV. The goal is to find a tour for the UAV that visits and covers all cells in minimum time. This includes flight time for visiting and covering all cells, recharging time, as well as the take-off and landing times. We show how to reduce this problem to a known NP-hard problem, Generalized Traveling Salesperson Problem (GTSP). Given an optimal GTSP solver, our approach finds the optimal coverage paths for the UAV and UGV. We evaluate our algorithm through simulations and proof-of-concept experiments.**

I. Introduction

Our work is motivated by applications such as infrastructure and environmental monitoring [1], [2], [3], surveillance [4], precision agriculture [5], [6], and search and rescue [7], [8] where Unmanned Aerial Vehicles (UAVs) are used as mobile sensors. However, most multi-rotor UAVs have limited battery capacities that restrict deployments to less than 30 minutes [9]. As a result, surveying large areas with a single vehicle may require frequent stops to recharge or replace batteries. Traveling to a fixed recharging station will further increase the time required to cover the environment. Instead of using fixed recharging stations, we show how to use an Unmanned Ground Vehicle (UGV) that acts as a mobile recharging station.

This work extends our prior work [10] wherein we presented a method to visit a set of sites using the UAV and the UGV as a mobile recharging station. In this paper, we extend it to handle the case where the input is not a set of sites but instead a set of regions that need to be covered.

The input to our planner is a set of *boustrophedon cells* — rectangular regions whose width is equal to the footprint of the UAV's sensor. The boustrophedon cells can be obtained by decomposing regions that need to be covered [11], [12]. Figure 1 shows a motivating example of surveying crops in four fields. Here, each boustrophedon cells corresponds to a row of crops that need to be covered by the UAV.

Yu and Tokekar are with the Department of Electrical & Computer Engineering, Virginia Tech, U.S.A. Email: {klyu, tokekar}@vt.edu
O'Kane is with the Department of Computer Science & Engineering, University of South Carolina, U.S.A. jokane@cse.sc.edu

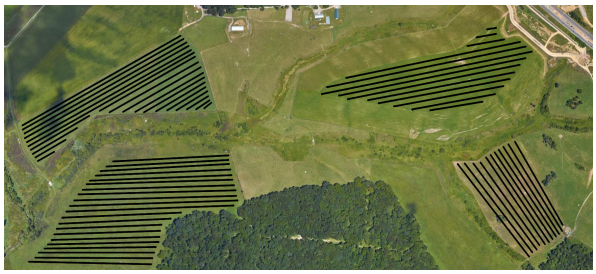


Fig. 1. We study the problem of covering a set of boustrophedon cells using a UAV which has limited battery capacity. In a precision agriculture scenario, a boustrophedon cell may correspond to a row of crops that must be monitored using a downward-facing camera on the UAV.

A boustrophedon cell can be covered by the UAV entering from either end and exiting from the other — the planner must find the optimal sequence in which to cover the boustrophedon cells as well as the corresponding entry and exit sites for each boustrophedon cells.

We consider scenarios where the UAV can land on the UGV and either recharge in-place or recharge while the UGV transports the UAV to the next take-off site. We present an algorithm that plans a tour for the UAV and a path for the UGV such that the UAV can cover an area in the minimum time while never running of out charge. This includes not only the flight time of the UAV but also the time it takes to recharge as well as the taking-off and landing times. The output tour given by our planner specifies not only the order in which to cover the boustrophedon cells but also the charging schedule. In particular, the planner outputs where to recharge the battery and how much to recharge by.

This problem is a generalization of the NP-hard Traveling Salesperson Problem (TSP) [13]. Our solution is based on reducing the coverage problem to that of the Generalized TSP (GTSP) [14]. Specifically, we present an algorithm that is guaranteed to find the optimal coverage tour for the UAV as long as there exists an optimal GTSP solver. While no polynomial-time optimal algorithms are believed to exist for NP-hard problems, there are solvers that find optimal solutions to many instances in reasonable amounts of time in practice. We use one such solver, Generalized Large Neighborhood Search (GLNS) [15], that finds optimal solutions for GTSP instances. We empirically evaluate the performance of our algorithm using the GLNS solver. We also demonstrate the efficacy of the system through proof-of-concept field experiments.

II. Related Work

Environmental coverage is a well-studied problem in robotics [16], [17]. The variant most closely related to the one we study is that of decomposing a known environment into various cells and then finding a route to sweep (i.e., cover) each cell [18], [19], [20], [21], [22]. In this work, we assume that the first step (cell decomposition) is solved and focus on the problem of routing the UAV to cover the cells in minimal time, while keeping track of the battery level. Specifically, we take as input a boustrophedon cell decomposition which can be found using techniques given by, for example, Maza and Ollero [12].

A number of algorithms have also been developed for the second step, i.e., routing to cover all cells, under various constraints. In particular, Karapetyan et al. [23] presented a multi-robot coverage algorithm for boustrophedon cell decompositions for point robots. Yu et al. [24] presented a coverage algorithm for a single robot, but with Dubins steering constraints. Lewis et al. [25] later proved that problem NP-hard, and showed how to reduce the graph to obtain practical solutions [26]. Bochkarev and Smith [22] present a method for decomposing an environment to minimize the number of turns needed to cover an area, but do not consider energy-limited robots and consequently, do not need to keep track of the battery level of the robot. Most recently, Karapetyan et al. [27] presented two techniques to cover a collection of cells with multiple Dubins vehicles.

The underlying ideas in the aforementioned works are similar — reduce the problem of covering all cells to a variant of the TSP, solve the TSP, and convert the resulting solution back to a tour for the robots. Our approach extends these ideas for the case of a robot with limited energy capacity which can be recharged along the way. This requires us to keep track of energy level of the robot along the tour which further complicates the problem. Nevertheless, we show that by reducing it to GTSP, we can obtain optimal solutions in reasonable amounts of time.

There have been algorithms for assigning and routing with one or more stationary recharging stations [28], [29], [30]. Kim et al. present a Mixed Integer Linear Programming approach for assigning UAVs to stationary recharging stations while taking into account the task objective [28]. Liu and Michael [30] presented a method for assigning UAVs to UGVs acting as recharging stations [30]. In our previous work [6], [10], we studied the problem of visiting a set of sites (points in a 2D plane) using an energy-limited UAV. In [6], we showed how to maximize the number of sites visited in a single charge when the UAV is allowed to land on the UGV and let the UGV transport it to the next take-off site without the UAV expending energy. In [10], we extended this to also allow for the UGV to recharge the UAV either while stationary or while being transported to the next deployment site. This paper extends the prior work from merely visiting sites to area coverage. As a result, the planner must decide not only the order in which the boustrophedon cells should be visited but also the directions in which to cover them.

III. Problem Formulation

The input to our algorithm is a set of n boustrophedon cells that need to be covered by the UAV. A boustrophedon cell is a rectangular strip whose width is equal to the diameter of the field-of-view (FOV) of the sensor onboard the robot. An example is shown in Figure 2 and a larger example is shown in Figure 1.

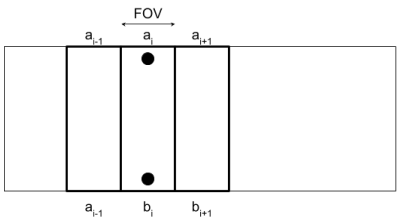


Fig. 2. Example boustrophedon cells. Each boustrophedon cell is a rectangle whose width is equal to the footprint of the UAV's sensor. A boustrophedon cell i is characterized by two sites, a_i and b_i, on either end. The algorithm must choose which one acts as the entry site.

Each boustrophedon cell consists of two sites, a_i and b_i, where i is the index of the boustrophedon cell. These sites are placed at two ends of the rectangular strip. A boustrophedon cell is said to be *covered* if the UAV traverses in a straight line from a_i to b_i or from b_i to a_i. The UAV can enter a boustrophedon cell from either site, a_i or b_i. However, once the UAV has entered a boustrophedon cell it is required to cover the entire boustrophedon cell and exit from the other site. The coverage algorithm must choose one of the sites as the entry site. We slightly generalize the traditional notion of a boustrophedon cell by allowing them to be oriented along different directions. That is, we do not require the boustrophedon cells to be parallel to each other.

We make the following assumptions:

- the UAV has an initial battery charge of 100%;
- the UAV flies at a fixed-altitude plane when covering a boustrophedon cell;
- the UAV travels at unit speed;
- the battery discharges at a unit rate (one unit per unit time and per unit distance traveled);
- the battery gets recharged at a rate of r units per unit time;
- the UGV has unlimited fuel/battery capacity.

Since we assume that the UAV travels with unit speed, we use time and distance interchangeably. We use t_{TO} and t_L to represent the time taken by the UAV to take-off from the UGV to reach the fixed-altitude plane and land from this plane onto the UGV, respectively. $D_{\max}$ represents the total distance a UAV can travel with 100% battery capacity.[1] We discretize the battery capacity into C levels.

Let $\gamma(i) \in \{a_i, b_i\}$ denote the site chosen by a coverage algorithm to be the entry site of the i^{th} boustrophedon cell. Correspondingly, $\overline{\gamma}(i)$ denotes the site chosen to be the exit

[1] Strictly speaking, we maintain a reserve battery capacity to take-off from ground to reach the fixed altitude plane and land from the fixed-altitude plane on the ground. In this paper, when we refer to 100% battery capacity, it excludes this reserve battery for taking-off and landing.

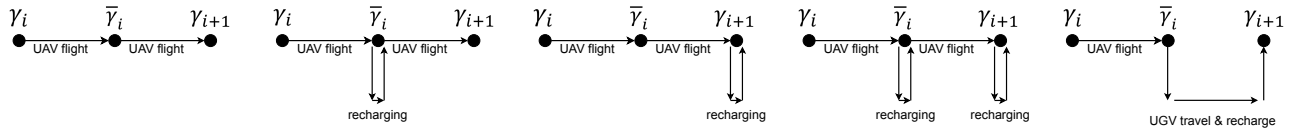


Fig. 3. Five edge types (F-F, F-DUF, F-FDU, F-DUFDU, F-DTU).

site of the boustrophedon cell, i.e., $\overline{\gamma}(i) = \{a_i, b_i\} \backslash \gamma(i)$. $\sigma(j)$ denotes the order in which the boustrophedon cells are to be visited. That is, $\sigma(j) \in \{1, \ldots, n\}$ gives the j^{th} boustrophedon cell that is visited.

We use γ_i, $\overline{\gamma}_i$, and γ_{i+1} to denote $\gamma(\sigma(i))$, $\overline{\gamma}(\sigma(i))$, and $\gamma(\sigma(i+1))$, respectively. We denote by $t_G(\overline{\gamma}_j, \gamma_{j+1})$ the time it takes for the UGV to travel from the exit site of j^{th} boustrophedon cell to the entry site of the next visited boustrophedon cell. We use $t_A(\overline{\gamma}_j, \gamma_{j+1})$ to represent the time it takes for the UAV to travel from the exit site of the j^{th} boustrophedon cell to the entry site of the next boustrophedon cell visited. Similarly, $t_A(\gamma_j, \overline{\gamma}_j)$ gives the time taken by the UAV to cover the j^{th} boustrophedon cell.

Suppose π is a path that visits every boustrophedon cell in the order given by σ and with entry and exit sites given by γ. The cost of the path depends on how the UAV travels between consecutive boustrophedon cells. Consider traveling from γ_j to $\overline{\gamma}_j$ and then on to γ_{j+1} along π. We have the following components for this part of the path:

- The UAV must fly from γ_j to $\overline{\gamma}_j$. The time taken is given by $t_A(\gamma_j, \overline{\gamma}_j)$.
- It can then choose to land on the UGV at $\overline{\gamma}_j$, recharge in-place, and take-off to reach the fixed-altitude plane at $\overline{\gamma}_j$. Let $I_1(\overline{\gamma}_j)$ be an indicator function that denotes whether the UAV chooses to do this or not.
- It can then choose to either fly from $\overline{\gamma}_j$ to γ_{j+1} or land on the UGV at $\overline{\gamma}_j$, recharge while being carried by the UGV to the next site, then take-off at γ_{j+1} to reach the fixed-altitude plane. Let $I_2(\overline{\gamma}_j)$ be an indicator function denoting whether the UAV travels with the UGV or not.
- It can then choose to land on the UGV at γ_{j+1}, recharge in-place, and take-off to reach the fixed-altitude plane at γ_{j+1}. Let $I_1(\gamma_{j+1})$ be an indicator function that denotes whether the UAV chooses to do this or not.

Based on these choices, the cost of traveling from γ_j to γ_{j+1} is given by:

$$\begin{aligned} T(j, j+1) &= t_A(\gamma_j, \overline{\gamma}_j) \\ &+ I_1(\overline{\gamma}_j)(t_L + r \cdot b(\overline{\gamma}_j, \overline{\gamma}_j) + t_{TO}) \\ &+ (1 - I_2(\overline{\gamma}_j)) t_A(\overline{\gamma}_j, \gamma_{j+1}) \\ &+ I_2(\overline{\gamma}_j)(\max\{t_G(\overline{\gamma}_j, \gamma_{j+1}), r \cdot b(\overline{\gamma}_j, \gamma_{j+1})\} + t_L + t_{TO}) \\ &+ I_1(\gamma_{j+1})(t_L + r \cdot b(\gamma_{j+1}, \gamma_{j+1}) + t_{TO}) \end{aligned} \tag{1}$$

Here, $b(\cdot)$ is a function which gives the amount by which the battery should be recharged between the two entry sites.

Therefore, we can define the cost of the path π as:

$$T(\pi) = \sum_{j=1}^{n-1} T(\gamma_j, \gamma_{j+1}) + t_A(\gamma_n, \overline{\gamma}_n) \tag{2}$$

We are now ready to define the problem.

Problem 1 (Multiple Polygon Coverage): Given a set of boustrophedon cells to be covered, find a path π^* for the UAV that visits and covers all of the boustrophedon cells, while minimizing the cost (Equation 2), and ensuring that the UAV does not run out of battery capacity. The path π^* must specify the order in which to visit the boustrophedon cells, $\sigma(\cdot)$, the entry site for each boustrophedon cell, $\gamma(\cdot)$, the recharging indicator functions, $I_1(\cdot)$ and $I_2(\cdot)$, and the amount of recharging at a site, $b(\cdot)$.

Note that finding a path for the UAV necessitates finding a path for the UGV that supports the UAV recharging schedule.

IV. GTSP-BASED ALGORITHM

We solve the polygon coverage problem by reducing it to GTSP. In this section, we describe in detail the reduction to GTSP. The input to GTSP is a graph where the vertices are partitioned into clusters. The objective is to find a minimum cost tour that visits exactly one vertex in each cluster. If all the clusters contain exactly one vertex, then GTSP trivially reduces to TSP. We show how to create the clusters, the edges between the clusters and then show how to convert the solution for GTSP into tours for the UAV and the UGV.

A. Vertices and Clusters

We discretize the battery into C levels. We create C vertices, one corresponding to each discretized battery level, for each site a_i and b_i corresponding to boustrophedon cell i. The vertex is denoted by $v_{a_i}^k$ or $v_{b_i}^k$, where k corresponds to the discretized battery level. We create one cluster per boustrophedon cell. This cluster contains $2C$ vertices, C of them corresponding to a_i and b_i each.

B. Edges

We create an edge between every pair of vertices that do not belong to the same cluster (i.e., do not belong to the same boustrophedon cell). Recall that a vertex corresponds to the entry site for the corresponding boustrophedon cell. Therefore, an edge between two vertices represents the UAV starting at the entry site of the first boustrophedon cell and ending at the entry site of the next boustrophedon cell. This includes two travel legs: coverage of the first boustrophedon cell and then traveling from the exit site of the first boustrophedon cell towards the second boustrophedon cell. Recall that the UAV must always fly the first leg; however, the second leg can be a combination of recharging, flying, and/or recharging while traveling on the UGV.

Equation 1 gives the cost of traveling between two entry sites of different boustrophedon cells. The actual cost depends on the three binary indicator variables: $I_1(\gamma_j)$,

$I_2(\overline{\gamma_j})$, and $I_1(\gamma_{j+1})$. This gives a total of 2^3 possible travel options. However, three of these eight options are redundant. Specifically, if the UAV chooses to recharge while traveling on the UGV, then also recharging on either end of this leg is redundant and, in fact, more time-consuming since it will have to take-off and land more than once. Formally, if $I_2(\overline{\gamma_j}) = 1$, then the optimal algorithm will never set $I_1(\gamma_j) = 1$ nor $I_1(\gamma_{j+1}) = 1$. Therefore, of these 2^3 possibilities, we can eliminate three, leaving a total of five possibilities for the second leg. These are shown in Figure 3. Note that since we assume that the UAV starts with 100% battery capacity, we will never recharge at the first entry site.

We denote the five edge combinations using the notation: F = Fly, D = Down/Land, U = Up/Take-off, and T = Transit. The first leg is always the UAV flying to cover the boustrophedon cell. We describe the actual edge costs in the remainder of this section.

In the following, we show how to compute the edge cost between vertices $v_{a_i}^{k_i}$ and $v_{a_j}^{k_j}$. k_i' denotes the battery at $v_{b_i}^{k_i'}$ if going from $v_{a_i}^{k_i}$ and $v_{a_j}^{k_j}$. Note that there will also be edges between $v_{b_i}^{k_i}$ and $v_{a_j}^{k_j}$, $v_{a_i}^{k_i}$ and $v_{b_j}^{k_j}$, and $v_{b_i}^{k_i}$ and $v_{b_j}^{k_j}$. The costs for these edges can be obtained using the same formula just by swapping a with b and vice-versa.

F-F*:* The cost of the F-F type of edge between $v_{a_i}^{k_i}$ and $v_{a_j}^{k_j}$ is ∞ if the energy required to go from a_i to b_i and then to a_j is more than $k_j - k_i$. Else, the edge cost is given by:

$$T_{\text{F-F}}(v_{a_i}^{k_i}, v_{a_j}^{k_j}) = t_A(v_{a_i}^{k_i}, v_{b_i}^{k_i'}) + t_A(v_{b_i}^{k_i'}, v_{a_j}^{k_j}). \tag{3}$$

F-FDU*:* The cost of the F-FDU type of edge between $v_{a_i}^{k_i}$ and $v_{a_j}^{k_j}$ is ∞ if the energy required to go from a_i to b_i and then to a_j is more than $k_j - k_i$. Else, the edge cost is:

$$\begin{aligned} &T_{\text{F-FDU}}(v_{a_i}^{k_i}, v_{a_j}^{k_j}) = \\ &t_A(v_{a_i}^{k_i}, v_{b_i}^{k_i'}) + t_A(v_{b_i}^{k_i'}, v_{a_j}^{k_j}) + t_L + r \cdot e + t_{TO}, \end{aligned} \tag{4}$$

where $e = \max\{0, k_j - (k_i - (\|a_i - b_i\|_2 + \|b_i - a_j\|_2))\}$ gives the recharging amount.

F-DUFDU*:* The cost of the F-DUFDU type of edge between $v_{a_i}^{k_i}$ and $v_{a_j}^{k_j}$ is equal to ∞ if the energy required to go from a_i to b_i is more than $k_i' - k_i$ and then b_i to a_j is more than $k_j - k_i'$. Else, the edge cost is given by:

$$\begin{aligned} &T_{\text{F-DUFDU}}(v_{a_i}^{k_i}, v_{a_j}^{k_j}) = \\ &t_A(v_{a_i}^{k_i}, v_{b_i}^{k_i'}) + t_L + r \cdot e_1 + t_{TO} \\ &+ t_A(v_{b_i}^{k_i'}, v_{a_j}^{k_j}) + t_L + r \cdot e_2 + t_{TO}, \end{aligned} \tag{5}$$

where $e_1 = \max\{0, k_i' - (k_i - \|a_i - b_i\|_2)\}$ and $e_2 = \max\{0, k_j - (k_i' - \|b_i - a_j\|_2)\}$ gives the recharging amount for e_1 and e_2, respectively.

F-DUF*:* The cost of the F-DUF type of edge between $v_{a_i}^{k_i}$ and $v_{a_j}^{k_j}$ is equal to ∞ if the energy required to go from a_i to b_i is more than $k_i' - k_i$ and then b_i to a_j is more than $k_j - k_i'$. Else, the edge cost is given by:

$$\begin{aligned} &T_{\text{F-DUF}}(v_{a_i}^{k_i}, v_{a_j}^{k_j}) = \\ &t_A(v_{a_i}^{k_i}, v_{b_i}^{k_i'}) + t_L + r \cdot e + t_{TO} + t_A(v_{b_i}^{k_i'}, v_{a_j}^{k_j}), \end{aligned} \tag{6}$$

where $e = \max\{0, k_i' - (k_i - \|a_i - b_i\|_2)\}$ gives the recharging amount.

F-DTU*:* The cost of the F-FDU type of edge between $v_{a_i}^{k_i}$ and $v_{a_j}^{k_j}$ is equal to ∞ if the energy required to go from a_i to b_i is more than $k_i' - k_i$. Else, the edge cost is given by:

$$\begin{aligned} &T_{\text{F-DTU}}(v_{a_i}^{k_i}, v_{a_j}^{k_j}) = \\ &t_A(v_{a_i}^{k_i}, v_{b_i}^{k_i'}) + t_L + \max(t_G(v_{b_i}^{k_i'}, v_{a_j}^{k_j}), r \cdot e) + t_{TO}, \end{aligned} \tag{7}$$

where $e = \max\{0, k_j - (k_i' - \|b_i - a_j\|_2)\}$ gives the recharging amount.

The actual edge cost between $v_{a_i}^{k_i}$ and $v_{a_j}^{k_j}$ is the minimum of all five types. Specifically, the final edge cost is given by: $T(v_{a_i}^{k_i}, v_{a_j}^{k_j}) = \min\{T_{\text{F-F}}, T_{\text{F-FDU}}, T_{\text{F-DUFDU}}, T_{\text{F-DUF}}, T_{\text{F-DTU}}\}$. We also keep track of which type of edge gives the final edge cost. This is used when converting the GTSP tour into a solution for the original problem.

C. Solving GTSP

We solve the GTSP instance using the GLNS solver [15]. GLNS uses a neighborhood search heuristic to find the optimal solution for the given GTSP instance. GLNS also allows for finding feasible solutions in lesser time, potentially at the expense of optimality.

Common alternatives for finding the optimal GTSP solution are Integer Programming or reducing GTSP to TSP [14] and then using a TSP solver such as Concorde [31]. In our previous work [10], we showed that GLNS finds the optimal solution for a similar class of GTSP instances in times that are at least an order of magnitude faster than the other approaches. As a result, we focus on only using GLNS for solving the GTSP instances in this paper.

D. Converting the GTSP solution to a Coverage Tour

The optimal tour obtained from the GTSP solver is a tour that visits exactly one vertex in each cluster. Recall that one cluster corresponds to one boustrophedon cell. The optimal tour will visit only one vertex within a cluster. The chosen vertex corresponds specifies the entry site for the boustrophedon cell as well as the corresponding battery level.

For example, if the edge between $v_{a_i}^{k_i}$ and $v_{b_j}^{k_j}$ is selected, then this implies the UAV will cover the i^{th} boustrophedon cell with a_i as the entry site and b_i as the exit site. Then, the UAV will travel from b_i to the entry site of the next boustrophedon cell which is chosen to be b_j. The actual mode of transportation between $v_{a_i}^{k_i}$ and $v_{b_j}^{k_j}$ depends on the type of the edge (F-F, F-FDU, F-DUFDU, F-DUF, or F-DTU). Depending the type, we construct the actual tour and recharging schedule for the UAV.

We can determine the UGV path based on the type of edges chosen by considering the edges in the order they appear in the optimal GTSP tour. For a F-F edge, the UGV is not required. For an F-DUF edge, we add the exit site of the first boustrophedon cell to the UGV tour. Similarly, for an F-FDU, we add the entry site of the second boustrophedon cell to the UGV tour. Finally, for F-DUFDU and F-DTU

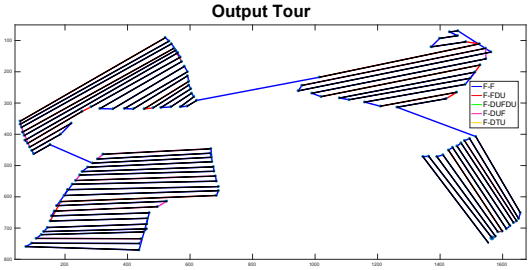


Fig. 4. Solution to the input instance given in Figure 1.

edges, we add the exit site of the first boustrophedon cell and the entry site of the second one to the UGV tour.

E. Performance Guarantees

If the GTSP solver finds the optimal tour, then the corresponding UAV tour is also the optimal solution to Problem 1 with the additional assumption that the UGV is as fast as the UAV. If the UAV is faster than the UGV, then it is possible that the solution yields paths where the UAV reaches a landing site before the UGV. In such cases, one possibility is to have more than one UGV that can support the UAV tour. In our previous work [32], we presented an Integer Programming solution that minimizes the number of slower UGVs required to support the UAV tour.

V. Simulations

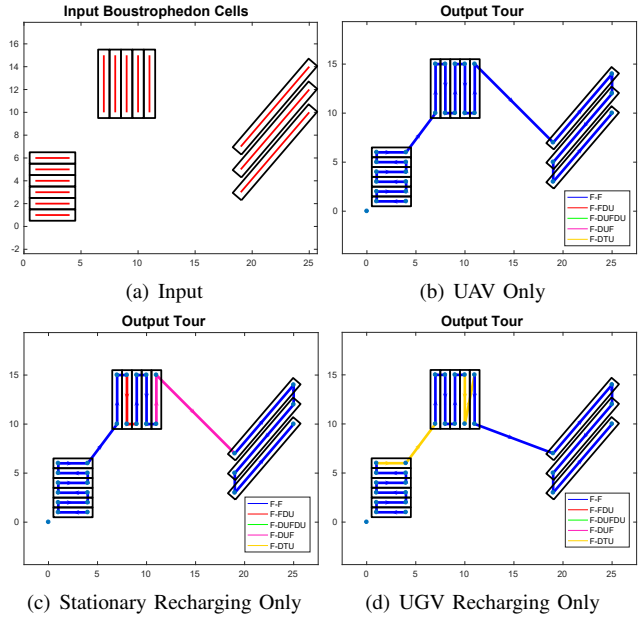
In this section, we present qualitative and quantitative results to evaluate the proposed algorithm. In particular, we analyze the effect of various parameters on the tour cost and the computational time of the algorithm. The experiments were run on an Ubuntu 14.04 computer with an Intel i7-6700HQ CPU running at 2.6GHz, with 4 cores, 16GB of RAM, and a GTX 980M GPU.

A. Qualitative Examples

We use the boustrophedon cells given in Figure 1 as the input. There are a total of 66 boustrophedon cells. The solution is shown in Figure 4. The boustrophedon cells are marked with rectangles. The input parameters were set to: $t_{TO} = 5$, $t_L = 45$, $r = 2$, $D_{\max} = 1800$ and $C = 20$. The UGV speed was set to be 20% as that of the UAV.

We compare the results of our algorithm with a naive baseline. The baseline approach visits each boustrophedon cells in the same order in which they appear along the boundary of the polygon. The UAV lands to recharge on the UGV only when covering the next boustrophedon cell would deplete it of the remaining energy. Once on the UGV, the UAV recharges to maximum capacity. The baseline approach produces a tour which requires 29564 seconds for completion (as given by Equation 2) where as the proposed algorithm produces a tour which requires 25910 seconds.

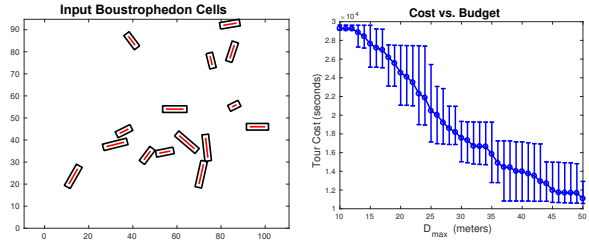
Figures 5(b)– 5(d) presents additional qualitative examples that show the effect of changing $D_{\max}$ and the UGV speed on the solution for the input given in Figure 5(a). We observe that if the UAV has enough energy capacity $D_{\max}$ then the final tour does not use the UGV (Figure 5(b)). If the UGV is



(a) Input (b) UAV Only

(c) Stationary Recharging Only (d) UGV Recharging Only

Fig. 5. (a) Input for qualitative examples to help explain input parameters and the effects. (b) Using input parameters: $t_{TO} = 5$, $t_L = 45$, $r = 2$, $D_{\max} = 200$, UGV speed is one-fifth of the UAV, and $C = 20$, results in a tour that uses only the UAV. (c) Minimum number of landings/take-offs in place for the given input parameters. The input parameters for this experiment were $t_{TO} = 5$, $t_L = 45$, $r = 2$, $D_{\max} = 50$ UGV speed is 50 times slower than the UAV, and $C = 20$. (d) Minimum number of landings/take-offs using the UGV to recharge for the given input parameters. The input parameters for this experiment were $t_{TO} = 5$, $t_L = 45$, $r = 2$, $D_{\max} = 50$ UGV speed is 5 times slower than the UAV, and $C = 20$.



(a) Input (b) Cost vs. budget

Fig. 6. (a) Example input boustrophedon cells for the 10 trials used for generating (b). We randomly create 15 non-overlapping boustrophedon cells, each no more than 10m. (b) Tour cost vs. flight budget, $D_{\max}$. We vary the total budget as well as the distance per battery level. The input parameters were: $t_{TO} = 1000$, $t_L = 1000$, $r = 2$, and $C = 20$. The UGV is 5 times slower than the UAV.

significantly slower than the UAV and $D_{\max}$ is small, then the UAV only recharges in-place (Figure 5(c)) and does not use the F-DTU edges. However, if the UGV is not as slow, then the tour uses F-DTU edges (Figure 5(d)). We present quantitative evaluation of these parameters next.

B. Effect of Changing $D_{\max}$ on the Tour Cost

Next, we study the effect of changing the total battery capacity, i.e., changing $D_{\max}$, on the total tour time. We randomly generate 15 boustrophedon cells in a 100m $\times$ 100m environment such that no two boustrophedon cells intersect with each other and each boustrophedon cell is no more than 10 meters long. Figure 6(a) shows one example.

We vary $D_{\max}$ from 10 meters to 50 meters. We use the same set of 15 boustrophedon cells for each value of $D_{\max}$. Figure 6(b) shows the average, minimum, and maximum value of the optimal tour cost. We observe that the tour cost decreases as $D_{\max}$ increases, as is expected. We also observe a step decrease in the minimum and maximum tour costs as $D_{\max}$ increases. This can be attributed to the fact that as $D_{\max}$ increases the UAV can travel a larger distance without running out of energy. Therefore, it may need to land/take-off fewer number of times. Each landing and taking-off operation costs a fixed amount of time. Therefore, we see a step decrease in the tour cost as $D_{\max}$ increases.

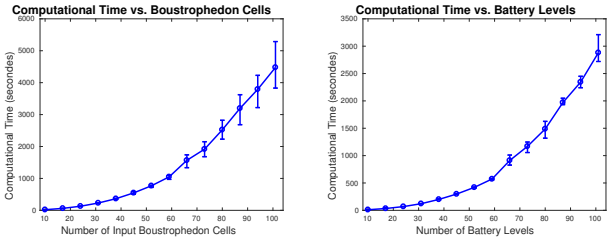
C. Effect on the Computational Time

Next, we empirically analyze the computational time as a function of some of the input parameters.

Figure 7(a) shows the effect of increasing the number of input boustrophedon cells on the computational time. The input number of boustrophedon cells are varied from 10 to 101 in steps of 7. The figure shows the average, minimum, and maximum computational times for 10 random instances. The input parameters for these experiments were kept the same: $t_{TO} = 100$, $t_L = 100$, $r = 2$, and $C = 20$.

Figure 7(b) shows the effect of increasing the battery level, i.e., C, on the computational time. We vary C from 10 to 101 in steps of 7. The input parameters for these experiments were: $t_{TO} = 100$, $t_L = 100$, $r = 2$. The figure shows the average, minimum, and maximum computational time for 10 random instances with 15 boustrophedon cells each.

We observe that the computational time increases (perhaps, exponentially) with increasing the number of input boustrophedon cells and battery level. Nevertheless, the computational time is still small enough (less than 50 minutes) for moderately sized instances (80 boustrophedon cells).

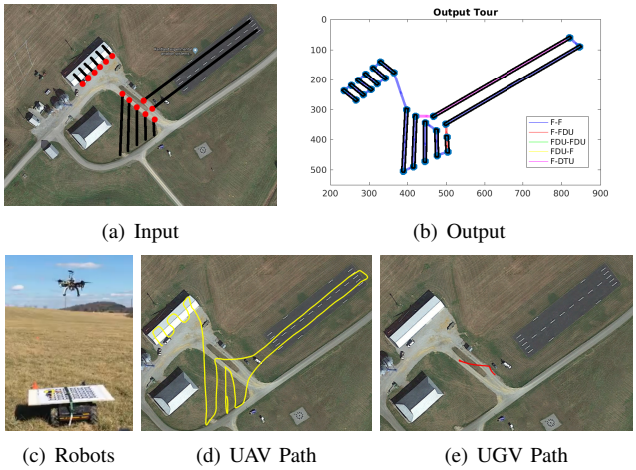


(a) (b)

Fig. 7. Input parameters: $t_{TO} = 100$, $t_L = 100$, $r = 2$, UGV speed is one-fifth that of the UAV for both plots. 10 random sets of input boustrophedon cells were randomly generated in a $100m \times 100m$ environment. We set $C = 20$ for (a) and vary C for (b).

VI. Field Experiments

We conducted proof-of-concept field experiments using the UAV and UGV shown in Figure 8(c). The UAV is a DJI 450 frame [33] with a Pixhawk 2.1 [34] flight controller running the APM firmware [35] and the UGV is a Clearpath Husky [36]. The UAV is equipped with dual GPSs, a downwards facing LIDAR (for relative altitude estimation) and the IR-Lock infrared camera [37]. The UGV is fitted with infrared LED beacons. The IR-Lock system [37] allows for precision landing on the UGV with up to 10cm accuracy in nominal wind conditions. More details on the system are reported in our prior work [32].



(a) Input (b) Output

(c) Robots (d) UAV Path (e) UGV Path

Fig. 8. Proof-of-concept Experiment with 13 boustrophedon cells. The input parameters were: $D_{\max} = 1000$, $C = 100$, $t_{TO} = 100$, $t_L = 100$, $r = 2$, and UGV speed is one-fifth that of the UAV. The UGV is also restricted to the road network (red sites).

Figure 8(a) shows the input boustrophedon cells for the proof-of-concept experiment conducted at Kentland Farms at Virginia Tech. The motion of the UGV is restricted to only those sites that lie on the road. Specifically, we allow F-FDU, F-DUF, F-DTU, or F-DUFDU edges only if the corresponding sites are on the road. These sites are marked in red in Figure 8(a). The output tour for the UAV is shown in Figure 8(b). The following parameters were used as input to the outdoor field experiments: $t_{TO} = 100$, $t_L = 100$, $r = 2$, $D_{\max} = 1000$, 13 boustrophedon cells, and $C = 100$. The GPS trace of the UAV and the UGV are shown in Figures 8(d) and 8(e). Additional experiments are included in the accompanying multi-media submission.

VII. Conclusion

We present an algorithm for optimal coverage of boustrophedon cells with an energy-limited UAV and a UGV. The UGV acts as a mobile recharging station that can mule the UAV between sites. We analyze the effects of various input parameters on the total tour cost as well as the computational time. We evaluate our algorithm through field experiments.

If the UGV is slower, it is possible that the UAV may reach a site before the UGV. In [32], we showed how find the minimum number of UGVs required to ensure that the UAV can execute its tour without having to wait for the UGV. A possible extension is to find a tour for a fixed number of slower UGVs that still ensures that the UAV does not need to wait for the UGV. We restrict the UAV to land and take-off only from the entry/exit sites of a boustrophedon cell. A possible extension would be to relax this assumption which can result in even shorter tours.

REFERENCES

[1] Peter Liu, Albert Y Chen, Yin-Nan Huang, J Han, J Lai, S Kang, T Wu, M Wen, and M Tsai. A review of rotorcraft unmanned aerial vehicle (uav) developments and applications in civil engineering. *Smart Struct. Syst*, 13(6):1065–1094, 2014.

[2] Tolga Ozaslan, Shaojie Shen, Yash Mulgaonkar, Nathan Michael, and Vijay Kumar. Inspection of penstocks and featureless tunnel-like environments using micro uavs. In *International Conference on Field and Service Robotics*, 2013.

[3] M. Dunbabin and L. Marques. Robots for environmental monitoring: Significant advancements and applications. *IEEE Robotics and Automation Magazine*, 19(1):24 –39, Mar 2012.

[4] Nathan Michael, Ethan Stump, and Kartik Mohta. Persistent surveillance with a team of mavs. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 2708–2714. IEEE, 2011.

[5] Jnaneshwar Das, Gareth Cross, Chao Qu, Anurag Makineni, Pratap Tokekar, Yash Mulgaonkar, and Vijay Kumar. Devices, systems, and methods for automated monitoring enabling precision agriculture. In *Proceedings of IEEE Conference on Automation Science and Engineering*, pages 462–469. IEEE, 2015.

[6] Pratap Tokekar, Joshua Vander Hook, David Mulla, and Volkan Isler. Sensor planning for a symbiotic UAV and UGV system for precision agriculture. *IEEE Transactions on Robotics*, 2016.

[7] Tristan Sherman, Joshua Tellez, Tristan Cady, Josue Herrera, Hana Haideri, Jimmy Lopez, Mitchell Caudle, Subodh Bhandari, and Daisy Tang. Cooperative search and rescue using autonomous unmanned aerial vehicles. In *2018 AIAA Information Systems-AIAA Infotech@ Aerospace*, page 1490. 2018.

[8] Yoonchang Sung and Pratap Tokekar. Algorithm for searching and tracking an unknown and varying number of mobile targets using a limited fov sensor. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 6246–6252. IEEE, 2017.

[9] 10 best long flight time drones: Fantastic battery life - 3d insider. https://3dinsider.com/long-flight-time-drones/. (Accessed on 09/21/2018).

[10] Kevin Yu, Ashish Kumar Budhiraja, and Pratap Tokekar. Algorithms and experiments on routing of unmanned aerial vehicles with mobile recharging stations. In *Proc. IEEE International Conference on Robotics and Automation*. IEEE International Conference on Robotics and Automation, 2018. To appear.

[11] Howie Choset and Philippe Pignon. Coverage path planning: The boustrophedon decomposition. In *Proceedings of the International Conference on Field and Service Robotics*, pages 3–91, 1997.

[12] Ivan Maza and Anibal Ollero. Multiple uav cooperative searching operation using polygon area decomposition and efficient coverage algorithms. In *Distributed Autonomous Robotic Systems 6*, pages 221–230. Springer, 2007.

[13] Sanjeev Arora. Polynomial time approximation schemes for euclidean traveling salesman and other geometric problems. *Journal of the ACM (JACM)*, 45(5):753–782, 1998.

[14] Charles E Noon and James C Bean. An efficient transformation of the generalized traveling salesman problem. *INFOR*, 31(1):39, 1993.

[15] S. L. Smith and F. Imeson. GLNS: An effective large neighborhood search heuristic for the generalized traveling salesman problem. *Computers & Operations Research*, 87:1–19, 2017.

[16] Howie Choset. Coverage for robotics - a survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31:113–126, 2001.

[17] Enric Galceran and Marc Carreras. A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12):1258–1276, 2013.

[18] W.H. Huang. Optimal line-sweep-based decompositions for coverage algorithms. In *Proc. the IEEE Int. Conf. on Robotics and Automation*, volume 1, pages 27 – 32, 2001.

[19] Z. Yao. Finding efficient robot path for the complete coverage of a known space. In *Proc. IEEE International Conference on Robotics and Automation*, 2006.

[20] Enrique Gonzalez, Oscar Alvarez, Yul Diaz, Carlos Parra, and Cesar Bustacara. BSA: a complete coverage algorithm. In *Proc. IEEE International Conference on Robotics and Automation*, 2005.

[21] Young-Ho Choi, Tae-Kyeong Lee, Sang-Hoon Baek, and Se-Young Oh. Online complete coverage path planning for mobile robots based on linked spiral paths using constrained inverse distance transform. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

[22] Stanislav Bochkarev and Stephen L Smith. On minimizing turns in robot coverage path planning. In *Automation Science and Engineering (CASE), 2016 IEEE International Conference on*, pages 1237–1242. IEEE, 2016.

[23] Nare Karapetyan, Kelly Benson, Chris McKinney, Perouz Taslakian, and Ioannis Rekleitis. Efficient multi-robot coverage of a known environment. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vancouver, BC, Canada, Sept. 2017.

[24] X. Yu, T. A. Roppel, and J. Y. Hung. An optimization approach for planning robotic field coverage. In *Proc. Annual Conference of the IEEE Industrial Electronics Society*, 2015.

[25] Kelly Benson Ioannis Rekleitis Jason M. O'Kane Jeremy S. Lewis, William Edwards. Semi-boustrophedon coverage with a dubins vehicle. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.

[26] Kelly Benson Ioannis Rekleitis Jason M. O'Kane Jeremy S. Lewis, William Edwards. Semi-boustrophedon coverage with a dubins vehicle. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.

[27] Nare Karapetyan, Jason Moulton, Jeremy S. Lewis, Alberto Quattrini Li, Jason M. O'Kane, and Ioannis Rekleitis. Multi-robot dubins coverage with autonomous surface vehicles. In *Proc. IEEE International Conference on Robotics and Automation*, 2018. To appear.

[28] Jonghoe Kim, Byung Duk Song, and James R Morrison. On the scheduling of systems of uavs and fuel service stations for long-term mission fulfillment. *Journal of Intelligent & Robotic Systems*, pages 1–13, 2013.

[29] Shaimaa Ahmed, Amr Mohamed, Khaled Harras, Mohamed Kholief, and Saleh Mesbah. Energy efficient path planning techniques for uav-based systems with space discretization. In *Wireless Communications and Networking Conference (WCNC), 2016 IEEE*, pages 1–6. IEEE, 2016.

[30] Lantao Liu and Nathan Michael. Energy-aware aerial vehicle deployment via bipartite graph matching. In *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, pages 189–194. IEEE, 2014.

[31] David Applegate, ROBERT Bixby, Vasek Chvatal, and William Cook. Concorde tsp solver. *URL http://www. tsp. gatech. edu/concorde*, 2006.

[32] Jfr_2018__mobile_charging-compressed.pdf. https://www.raas.ece.vt.edu/wordpress/wp-content/uploads/2018/09/JFR_2018__Mobile_Charging-compressed.pdf. (Accessed on 09/21/2018).

[33] Amazon.com: Dji flame wheel f450 arf kit: Camera & photo. https://www.amazon.com/DJI-Flame-Wheel-F450-ARF/dp/B00G4A2RBU. (Accessed on 09/19/2018).

[34] Pixhawk2.1 standard set - pixhawk2. http://www.proficnc.com/system-kits/31-pixhawk2-suite.html. (Accessed on 09/19/2018).

[35] Ardupilot firmware : /copter. http://firmware.ardupilot.org/Copter/. (Accessed on 09/19/2018).

[36] Husky ugv - outdoor field research robot by clearpath. https://www.clearpathrobotics.com/husky-unmanned-ground-vehicle-robot/. (Accessed on 09/14/2018).

[37] Ir-lock — infrared tracking systems for drones and robot automation. https://irlock.com/. (Accessed on 09/14/2018).