Persistent Monitoring With Refueling On A Terrain Using A Team Of Aerial And Ground Robots

Parikshit Maini, Kevin Yu, P. B. Sujit, and Pratap Tokekar

***Abstract*— There are many applications such as surveillance and mapping that require persistent monitoring of terrains. In this work, we consider a heterogeneous team of aerial and ground robots that are tasked with monitoring a terrain along a given path. Both types of robots are equipped with cameras that can monitor the terrain within their fields-of-view. We also consider the ability of the aerial robots to land occasionally on the terrain to recharge. The objective is to find a path for all the robots to reduce the time required. Determining optimal routes for the robots is a challenging problem because of constrained visibility due to the terrain and fuel limitations of the robots. We devise an MILP formulation for the problem using a 1.5 dimensional representation model. A branch-and-cut framework is used to implement the MILP and involves the design of a separation algorithm to compute valid inequalities. We report results from extensive simulations and proof-of-concept field experiments to show the efficacy of our approach.**

I. Introduction

Vision-based persistent monitoring is a requirement in many applications such as search-and-rescue [1], environmental monitoring [2] and precision agriculture [3]. In particular, there is significant interest in using Unmanned Aerial Vehicles (UAVs) for this task. A major limitation, however, is the limited battery lifetime of UAVs. This task becomes even more challenging when the geometry of the terrain to be monitored is complicated and leads to line-of-sight obstructions. The fuel capacity of the vehicles and consequent refueling must also be taken into account while planning for persistent missions. In a previous work [4], on visibility based monitoring of a path using a heterogeneous robot team, we developed techniques to account for visibility restrictions along a path, imposed due to the terrain. In this paper, we focus on computing optimal plans (paths) for a team of robots consisting of UAVs and Unmanned Ground Vehicles (UGVs) for a persistent monitoring scenario along linearly-approximable features within a terrain. A sample instance of the problem is shown in Figure 1.

In a persistent mission, the robot path can either be an infinite walk or a tour that the robot keeps traversing repeatedly. We address the latter case wherein the robot must repeat a certain tour in the environment. Such a tour is also referred to as the *kernel* [5] of an infinite mission. Note that the kernel can visit certain locations multiple times. We restrict ourselves to compute kernels that may visit a location a maximum of κ number of times. The terms tour, kernel and route are used interchangeably through the rest of the text.

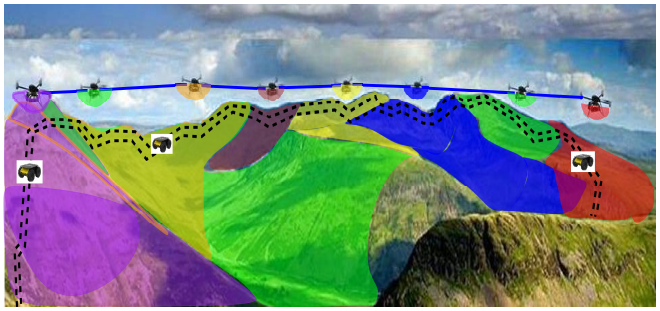


Fig. 1: Dashed lines show a linearly-approximable feature on the terrain. Visible area on the terrain is color coded for each robot location.

For an infinite monitoring scenario using robots, refueling is an important consideration. We need to ensure that the robots do not run out of fuel. In this work, we consider fuel limitations only for the UAV because, UAVs have a strict payload constraint and thus can carry only a limited amount of fuel. Furthermore, refueling incurs a large cost in the case of a UAV.

II. Related work

The problem generalizes the Watchman Route Problem (WRP) [6] to terrains and is an extension to the heterogeneous robotic sensors version [4] for monitoring paths on a terrain. Efrat et al. [7] studied the problem of sweeping a 2.5D terrain so as to detect mobile adversarial agents. They present an algorithm for n homogeneous aerial robots, that guarantees that every point on the terrain will be eventually seen by a robot. Under restrictions that the paths of the robots are restricted to straight lines, they present an exact polynomial time algorithm. Encouraged by recent results that show a constant-factor approximations can be computed for the n-homogeneous-WRP [8] [9] in *street polygons*, a special class of 2D polygons without holes, we solved the diverse robotic sensors case in [4]. In this work, we extend our results in [4] to address the persistent monitoring problem on a linearly-approximable feature, for instance boundaries, roads etc., within a terrain. Fuel constraints of the UAV impose severe limitations on the routing plans in this scenario. We solve the coupled problem on computing watchman routes for the robots in a persistent mission and ensuring that the aerial robots do not run out of fuel.

P. Maini and P. B. Sujit are with the Indraprastha Institute of Information Technology, India. {parikshitm, sujit}@iiitd.ac.in.

K. Yu and P. Tokekar are with the Department of Electrical & Computer Engineering, Virginia Tech, USA. {klyu, tokekar}@vt.edu.

This material is based upon work supported by the National Science Foundation under Grant #1566247 and EPSRC Grant EP/P02839X/1. Maini's research is supported by TCS Research Scholarship.

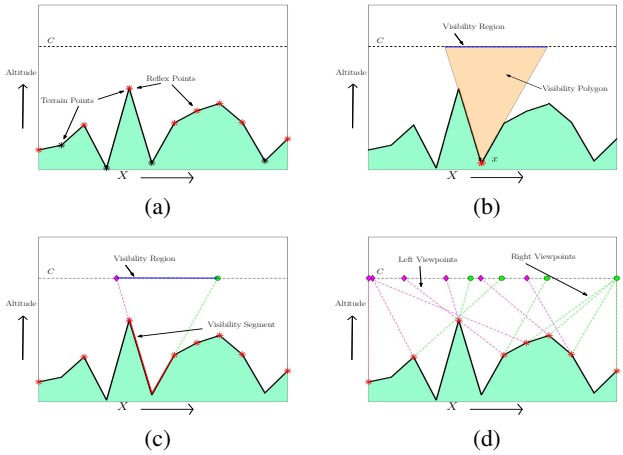


Fig. 2: (a) Terrain points, reflex points and a fixed altitude path chain visible with the terrain. (b) Visibility polygon and visibility region corresponding to a point x on the terrain. (c) A visibility segment and its corresponding visibility region. (d) Left and right viewpoints for all visibility segments.

Fuel constrained vehicle routing, also referred to as energy constrained routing has received a lot of attention over the past decade. A single vehicle fuel-constrained routing problem for a ground vehicle for a given set of stationary refueling stations was proposed in [10]. Mixed Integer Linear Programming (MILP) approaches and heuristics to solve the fuel constrained routing for a UAV have been proposed in [11]–[14]. Sundar et. al. [11] [12] have developed MILP based exact methods for UAV routing with stationary refueling stations. They present polyhedral analysis for the solution space and develop approximation algorithm and heuristics for the multiple refueling stations version. Maini et. al. [13] [14] address the mobile refueling station version in the context of coverage applications. We address the heterogeneous vehicle routing version with stationary refueling stations for persistent monitoring for a linearly-approximable feature within a terrain. We develop an MILP formulation and implement it using a branch-and-cut framework that uses runtime callback technique to dynamically add valid inequalities.

III. Preliminaries

A 1.5D terrain can be interpreted as a function $f(x)$ that returns an altitude value for every x along a single dimension. A linearly-approximable feature within a terrain can be *straightened* out to create a 1.5D representation. This admits the usage of *chain visibility* property of 1.5D terrains and fixed altitude paths.

We use the concept of chain visibility as defined in [4] for a curve C and a set of points X in a 2D space. Chain visibility has been used earlier in [8] and [15]. Some examples of chain visible pairs include street polygons and collapsed watchman routes [15], points on a 1.5D terrain and fixed altitude paths [8]. We refer the interested reader to [4] to read more about chain visibility. In our case, the fixed altitude paths corresponding to UAV flight altitudes and the set of point on the terrain form a chain visible pair. Also, a curve is chain visible to any set of points that lie on the curve itself, hence the terrain (UGV path) forms a chain visible pair with the points on the terrain.

We use the terminology for terrain features as defined in [4] to develop a discrete representation that accounts for visibility constraints of the environment. We show an illustration of the various features in Figure 2 and give a brief overview here, for completeness. Points within the terrain with different slopes of incoming and outgoing edges are called *terrain points* (Figure 2a). The intersection of visibility polygon of a point on the terrain and the fixed altitude UAV flight path, is called the visibility region of the point (Figure 2b). The subset of terrain points at which the slope of the outgoing edge is less than the slope of the incoming edge are called *reflex points* (Figure 2a). Two consecutive reflex points on the terrain mark a convex polyline called a *visibility segment* (Figure 2c). Extended rays along the outgoing and incoming edges at a reflex point (towards the reflex point) meet the fixed altitude paths at the left and right viewpoints corresponding to the reflex point, respectively. In case of visibility obstructions due to the terrain , the viewpoints are suitably adjusted (Figure 2d). These definitions are extended to the visibility region of a visibility segment on the terrain. The left viewpoint of the left reflex point of a visibility segment marks the left viewpoint of the visibility region of the entire segment. Similarly definition holds for the right viewpoint.

IV. Application Scenario

Consider an environment $\mathcal{E}$ (Figure 1) with a linearly approximable feature represented as a 1.5D terrain and a set of robots comprising both ground and aerial robots. The aerial robots are restricted to fly at constant altitudes. Each set of aerial robots flying at a given altitude contribute a chain visible curve in addition to the one on the terrain that corresponds to the ground vehicles. Let $\mathcal{C}$ be the set of chain visible curves corresponding to fixed-altitude UAV paths and $\mathcal{A}_c$ be the set of autonomous robots, ground or aerial, on the curve c.

The set of reflex points, R, and terrain end points comprise the set $\mathcal{R}$. ν is the set of visibility segments on the terrain. Also, by the chain visibility property all visibility segments are visible from each curve in $\mathcal{C}$. Let $\mathcal{V}_c^L$ and $\mathcal{V}_c^R$ be the set of left and right viewpoints on curve $c \in \mathcal{C}$ (Figure 2). Let $\mathcal{V}_c = \mathcal{V}_c^L \bigcup \mathcal{V}_c^R$, be the set of all viewpoints on the curve c and $\mathcal{V} = \bigcup_{c \in \mathcal{C}} \mathcal{V}_c$ be the set of all viewpoints. $\mathcal{V}^L = \bigcup_{c \in \mathcal{C}} \mathcal{V}_c^L$ and $\mathcal{V}^R = \bigcup_{c \in \mathcal{C}} \mathcal{V}_c^R$ are defined similarly. The sets $\mathcal{V}_c(v)$, $\mathcal{V}_c^L(v)$, $\mathcal{V}_c^R(v)$, $\mathcal{V}^L(v)$ and $\mathcal{V}^R(v)$ are subsets of viewpoints corresponding to the visibility segment v contained in the respective set.

In addition, there exist a set of refueling depots, $\mathbb{D}$, located on the terrain. Let $\mathbb{D}_c$ be the set of points closest to the points in $\mathbb{D}$ on the curve $c \in \mathcal{C}$. For each constant altitude flight path (chain visible with the terrain), the points in $\mathbb{D}_c$ lie at the intersection of curve c and the line perpendicular to c passing through the corresponding point in $\mathbb{D}$. Let U

be the maximum fuel capacity of the aerial robot. It may refuel at any of the refueling depots, as needed. Assuming a constant rate of fuel consumption regardless the maneuver and a constant speed for each robot; flight time, distance traveled, fuel consumed and cost are proportional quantities and are used interchangeably.

Our goal is to find routes for each robot to ensure that every point on the terrain is visible from at least one robot route. We restrict the *kernel* for each robot to visit a viewpoint a maximum of κ number of times. It may be observed that in an optimal solution for a general instance of a 1.5D terrain κ would not be larger than 2. In such a scenario, the following Fuel Constrained Heterogenous Watchman Routing Problem on a Terrain (FCHWRPT) arises naturally: *Given a 1.5D terrain and a set of ground and aerial robots with optical sensors and fuel limitations for the aerial robots, plan routes for the robots such that all points in the terrain are observed by at least one robot and the maximum path cost for a single robot is minimized over all robots.*

V. Solution Approach

We devise a Mixed Integer Linear Program (MILP) implemented within a branch-and-cut framework to compute optimal solution to FCHWRPT. To address the visibility restrictions, we define a visibility function, γ, and a cost function, α, on the viewpoints to account for travel costs. The visibility function, $\gamma_c(v) : \nu \rightarrow \wp(\mathcal{V}_c)$, where $\wp(x)$ is the power set of x, is defined for each curve $c \in \mathcal{C}$. $\gamma_c(v)$ is the set of viewpoints on curve c that lie between the left and right end points of the visibility region of visibility segment v on c, thus $\gamma_c(v) \subseteq \mathcal{V}_c$. Also, $\gamma(v) = \bigcup_{c\in\mathcal{C}} \gamma_c(v)$. The cost function, $\alpha^c_{ij} : V_c \times V_c \rightarrow \mathbb{R}^+$, returns the cost to travel from i to j on the chain visible curve c, where i and j belong to the set $V_c = \mathcal{V}_c \bigcup \mathbb{D}_c$.

A. Formulation

We define binary decision variables, $x^{ckl_1l_2}_{ij}$, for each $c \in C, k \in \mathcal{A}_c, i, j \in V_c$ and $l_1, l_2 \in \{1 \dots \kappa\}$. $x^{ckl_1l_2}_{ij} = 1$, if the k^{th} vehicle on the c^{th} chain visible curve visits the j^{th} viewpoint along the edge (i, j) on c and 0 otherwise. l_1 and l_2 refer to the current visit count by the k^{th} vehicle to viewpoints i and j, respectively. For each $c \in C$ and $\mathcal{P} \subseteq \mathcal{V}_c$, let $\delta^+_c(\mathcal{P}) \equiv \{(i, j) : i \in \mathcal{P}, j \in \mathcal{V}_c \setminus \mathcal{P}\}$, be the set of all outgoing edges from set $\mathcal{P}$ to $\mathcal{V}_c \setminus \mathcal{P}$. $z^{ckl_1l_2}_{ij}$ are real valued decision variables that represents the amount of fuel consumed by the k^{th} vehicle on the c^{th} chain visible curve to visit the j^{th} viewpoint along the edge (i, j) on c, from the most recently visited depot. l_1, l_2 refer to the current visit count by the k^{th} vehicle to the two viewpoints. y^{ck}_i are binary variables that take value 1 if the k^{th} vehicle on the c^{th} chain visible curve visits the i^{th} vertex, and 0 otherwise.

We consider the objective of the MILP to minimize the maximum path cost over all vehicles.

$$\text{minimize} \max_{c\in C,\ k\in\mathcal{A}_c} \sum_{l_1\in\{1\dots\kappa\}} \sum_{l_2\in\{1\dots\kappa\}} \sum_{i\in V_c} \sum_{j\in V_c} \alpha^c_{ij} \cdot x^{ckl_1l_2}_{ij}$$

The constraints used in the MILP are as follows:

Degree Constraints:

$$\sum_{l_2\in\{1\dots\kappa\}} \sum_{j\in V_c\setminus i} x^{ckl_2l_1}_{ji} - \sum_{l_2\in\{1\dots\kappa\}} \sum_{j\in V_c\setminus i} x^{ckl_1l_2}_{ij} = 0, \quad \forall c \in \mathcal{C}, k \in \mathcal{A}_c, l_1 \in \{1 \dots \kappa\}, i \in V_c \tag{1}$$

$$\sum_{l_2\in\{1\dots\kappa\}} \sum_{j\in V_c\setminus i} x^{ckl_2l_1}_{ji} \leq 1, \quad \forall c \in \mathcal{C}, k \in \mathcal{A}_c, l_1 \in \{1 \dots \kappa\}, i \in \mathcal{V}_c \tag{2}$$

$$\sum_{l_3\in\{1\dots\kappa\}} \sum_{j\in V_c\setminus i} x^{ckl_3l_1}_{ji} - \sum_{l_3\in\{1\dots\kappa\}} \sum_{j\in V_c\setminus i} x^{ckl_3l_2}_{ji} \geq 0, \quad \forall c \in \mathcal{C}, k \in \mathcal{A}_c, l_1, l_2 \in \{1 \dots \kappa\}, l_2 = l_1 + 1, i \in V_c \tag{3}$$

Visit Constraints:

$$y^{ck}_i - \sum_{l_2\in\{1\dots\kappa\}} \sum_{j\in V_c\setminus i} x^{ckl_21}_{ji} = 0, \quad \forall c \in \mathcal{C}, k \in \mathcal{A}_c, i \in \mathcal{V}_c \tag{4}$$

$$y^{ck}_i - x^{ckl_21}_{ji} \geq 0, \quad \forall c \in \mathcal{C}, k \in \mathcal{A}_c, i \in \mathbb{D}_c, j \in V_c \setminus i, l_2 \in \{1 \dots \kappa\} \tag{5}$$

$$y^{ck}_i - \sum_{l_2\in\{1\dots\kappa\}} \sum_{j\in V_c\setminus i} x^{ckl_21}_{ji} \leq 0, \quad \forall c \in \mathcal{C}, k \in \mathcal{A}_c, i \in \mathbb{D}_c \tag{6}$$

Sub-tour Elimination Constraints:

$$\sum_{l_1\in\{1\dots\kappa\}} \sum_{l_2\in\{1\dots\kappa\}} \sum_{(i,j)\in\delta^+_c(\mathcal{P})} x^{ckl_1l_2}_{ij} \geq 1 + \sum_{i\in\mathcal{P}} (y^{ck}_i - 1), \quad \forall c \in C, k \in \mathcal{A}_c, \mathcal{P} \subseteq V_c \setminus s^k_c \tag{7}$$

Coverage Constraints:

$$\sum_{c\in C} \sum_{k\in\mathcal{A}_c} \sum_{i\in\gamma_c(v)} y^{ck}_i \geq 1, \quad \forall v \in \nu \tag{8}$$

Refueling Constraints:

$$\sum_{l_2\in\{1\dots\kappa\}} \sum_{j\in V_c} z^{ckl_1l_2}_{ij} - \sum_{l_2\in\{1\dots\kappa\}} \sum_{j\in V_c} z^{ckl_2l_1}_{ji} = \sum_{l_2\in\{1\dots\kappa\}} \sum_{j\in V_c} \alpha^c_{ij} \cdot x^{ckl_1l_2}_{ij}, \quad \forall c \in C \setminus c_0, k \in \mathcal{A}_c, i \in \mathcal{V}_c, l_1 \in \{1 \dots \kappa\} \tag{9}$$

$$z^{ckl_1l_2}_{di} = \alpha^c_{di} \cdot x^{ckl_1,l_2}_{di}, \quad \forall c \in C \setminus c_0, k \in \mathcal{A}_c, l_1, l_2 \in \{1 \dots \kappa\}, d \in \mathbb{D}_c, i \in V_c \tag{10}$$

$$z^{ckl_1l_2}_{ij} \leq (U - \min_{d\in\mathbb{D}_c} \alpha^c_{jd}) \cdot x^{ckl_1l_2}_{ij}, \quad \forall c \in C \setminus c_0, k \in \mathcal{A}_c, l_1, l_2 \in \{1 \dots \kappa\}, i, j \in V_c \tag{11}$$

Variable Domain Constraints:

$$x^{ckl_1l_2}_{ij} \in \{0, 1\}, \quad \forall c \in C, k \in \mathcal{A}_c, l_1, l_2 \in \{1 \dots \kappa\}, i, j \in V_c \tag{12}$$

$$y_i^{ck} \quad \in \{0,1\}, \forall c \in C, k \in \mathcal{A}_c, i \in V_c \tag{13}$$

$$z_{ij}^{ckl_1,l_2} \quad \in [0,U], \quad \forall c \in C \setminus c_0, k \in \mathcal{A}_c, l_1, l_2 \in \{1 \dots \kappa\}, i,j \in V_c \tag{14}$$

Constraint 1 is a set of degree constraints that ensures that the result is a closed walk. Constraints 2 and 3 together ensure that no viewpoint is visited more than κ number of times. Constraints 4, 5 and 6 populate the visit variables for viewpoints and depots. Constraint 7 ensure that there does not exist a disconnected subtour not reachable from the starting point. Constraint 8 ensures that for each visibility segment v at least one viewpoint in the set $\gamma(v)$ is visited by one of the vehicles, thus ensuring coverage. Constraints 9 and 10 ensure fuel conservation at the viewpoints and when traveling out of a depot, respectively. Constraint 11 makes sure the aerial robot has enough fuel to reach the nearest depot. Constraints 12, 13 and 14 specify the domain for decision variables used in the formulation. As is clearly visible, the number of sub-tour elimination constraints is exponential in the number of viewpoints.

Algorithm 1 Separation Algorithm

```
for all c ∈ C do
    for all k ∈ A_c do
        Build graph G(directed) ≡ (V = V_c, E)
        Add edge (i, j) to E, if ∃_{l1,l2∈{1...κ}} x_ij^{ckl1l2} = 1
        Find connected components in G
        for all connected components G ∈ G do
            if (|G| > 1)&&(G ⊆ V_c \ s_c^k) then
                Add violation constraint (Eq. (7))
```

B. Branch-and-Cut Algorithm

As the number of sub-tour elimination constraints is exponential and it is not computationally efficient to enumerate them all and add to the solver. To address this issue, we use a branch-and-cut framework that works as follows: we solve a relaxed version of the problem that does not include the sub-tour elimination constraints. When the solver obtains an integer feasible solution to this relaxed problem, we check if the feasible solution violates any of these constraints. If so, we add the violated constraints to the formulation and continue solving the problem. This process of adding constraints to the problem sequentially has been observed to be computationally efficient for many variants of the Traveling Salesman Problem [16] and fuel constrained vehicle routing problems [11]. The algorithm to identify violated constraints is called a separation algorithm. We present psuedocode for a separation algorithm in Algorithm 1. The procedure computes the connected components in the graph defined by the integer solution for each vehicle. Each component P of cardinality greater than one that satisfies the condition $P \subseteq \mathcal{V}_c \setminus \{s_c^k\}$, where c is the curve on which the k^{th} vehicle traverses and s_c^k is the starting point of the vehicle, violates the corresponding constraint (Constraint 7). The violated constraints are then added to the formulation and the solver is allowed to optimize the problem with the new constraints.

VI. Simulation Results

A. Simulation Setup

Synthetic 1.5 dimensional terrains were generated for the simulations. The size of the terrain was varied, in terms of terrain points, from 5 to 25 in steps of 5. 50 random environments were generated of each size. The cost function used for the ground robot was asymmetric and dependent on the slope of the terrain on the path. The aerial robot cost function is directly proportional to the distance traveled, both for fixed altitude flight and for the take off-landing sequences. The maximum flight length/fuel capacity of the aerial robot, in terms of the flight cost, was varied from 1500 to 6000 units in steps of 1500.

The simulations were conducted using one ground robot and one aerial robot. Refueling depots were placed at alternative reflex points. Starting locations for the two robots were generated randomly. For the UAV, the starting location was selected from amongst one of the depots. For the UGV, the starting location was randomly selected from the set of reflex points in the terrain. Figure 3 shows simulation results for a sample instance for different U values.

Size	U = 1500	U = 3000	U = 4500	U = 6000	U = 7500
5	50	50	50	50	50
10	50	50	50	50	50
15	50	48	43	44	50
20	50	24	16	15	16
25	50	10	4	5	4

TABLE I: Number of instances (out of 50 random instances) solved optimally by the MILP solver.

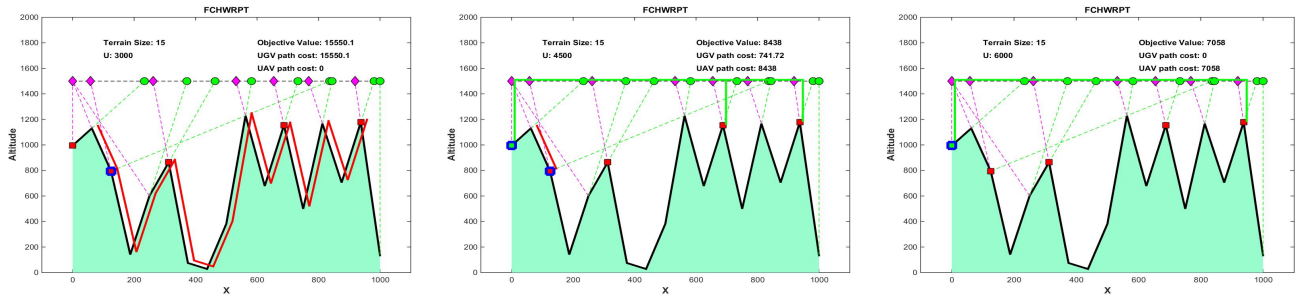
Size	U = 1500	U = 3000	U = 4500	U = 6000	U = 7500
5	0.02	0.09	0.14	0.16	0.19
10	0.27	0.93	1.84	2.26	2.36
15	0.68	12.05	50.82	57.63	26.63
20	5.84	45.20	188.44	298.52	206.29
25	26.80	101.64	229.10	405.51	444.96

TABLE II: Average computation time in seconds.

Size	U = 1500	U = 3000	U = 4500	U = 6000	U = 7500
5	3440	2988	2692	2657	2608
10	7801	6587	4844	3996	3699
15	11725	9450	7025	5128	4712
20	17078	13353	4442	3926	4158
25	21781	17159	3069	3538	3069

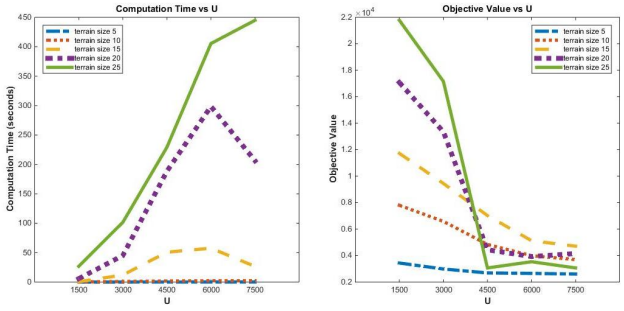
TABLE III: Average objective function value in terms of maximum path cost / fuel consumed by a single robot.

The MILP was implemented in C++ using IBM ILOG CPLEX library version 12.7 within a branch-and-cut framework. We utilized the lazy constraint callback functionality



(a) (b) (c)

Fig. 3: Simulation results for a sample instance. Red squares represent depots. UAV starting location is shown using double square of blue and cyan color. UGV starting location is shown using double square of blue and red color. UAV and UGV paths are shown in solid green and red lines, respectively. (a) Optimal solution uses only UGV for low U value. (b) For medium U values the solution uses both robots. (c) Optimal solution uses only UAV for high U value.



(a) (b)

Fig. 4: (a). Average computation time vs U value plot. (b) Average objective function value vs U value plot.

of CPLEX library to implement the separation algorithm for dynamic constraint addition. This was used to add the sub-tour elimination constraints at runtime to the formulation, thus allowing the solver to initiate the optimization process using the relaxed formulation for FCHWRPT.

B. Results and Analysis

The MILP formulation, though exponential in the number of constraints, performs well in a branch-and-cut framework. The addition of constraints during run time, allows it to scale to instances of large size. The simulations were conducted for a total of 1250 instances and each instance was allowed to run for a maximum of 15 minutes. Tables I, II and III summarize the simulation results. Tables II and III report the results only for optimally solved instances as mentioned in Table I. As may be observed from Table I, the number of instances optimally solved decrease with increase in size of the instance and the value of U. This is expected behavior. However, since this is an offline planning problem, we can afford large computation time. Due to the given time limit (15 minutes) only a few instances of the largest environment size (25 terrain points) were solved optimally. The number is too less to comment on general trends. We report results for instances with 25 terrain points but do not make inferences based on these.

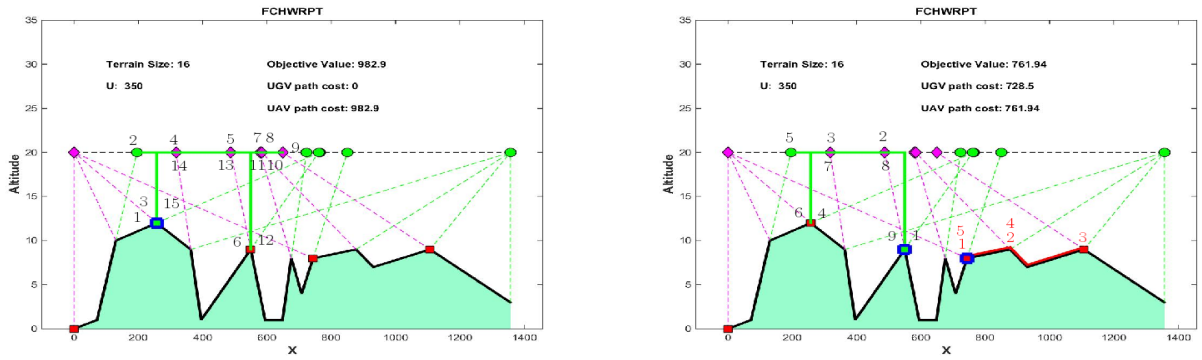
Table II shows the computation time taken by CPLEX MILP solver to find the optimal solution averaged over multiple environments. The values show a clear trend of large increase in the computational time with increase in the size of the environment (Figure 4a). Figure 4b shows that the objective function attains a stable value after a certain threshold of U value (Table III gives the objective function value for different configurations).

VII. Field Experiments

We evaluated our algorithm in field experiments. We used a custom made quadcopter UAV built on a DJI Flame Wheel F450 frame with Pixhawk 1 flight controller running APM V3.5.4 firmware. The UAV comprises of an externally mounted GPS sensor and an on board computer (Nvidia Jetson TX1). For the UGV, we used a custom modified Clearpath Husky (Figure 6a). When conducting experiments we launch both UAV and UGV at the same time. Both start at their respective start points and then move independently to their designated waypoints, assigned by the algorithm. The UAV was operated at an altitude of 20 meters. For the purpose of the experiment a linearly approximable *feature*, as shown in Figure 6b, was used. Flight range of the UAV, U, was fixed at 350 meters. A virtual simulated terrain was superimposed on the given *feature*. Figure 5 shows the 1.5 D representation of the terrain. It also show the path generated by the solver for the two experiments. The refueling depots, shown as red squares in Figure 5, were placed at alternate reflex points within the terrain. The refueling process was simulated using a land and take-off sequence during the experiment. We show the results for our experiments in the multi-media submission.

VIII. Conclusions

In this paper, we have addressed the problem of persistent monitoring of linearly-approximable features within a terrain using a heterogeneous team of UAVs and UGVs. We map the problem to a fuel constrained heterogeneous watchman routing in 1.5D terrains (FCHWRPT). This problem is a combination of set cover, vehicle routing and persistent monitoring. We developed an MILP formulation to solve



(a) (b)

Fig. 5: Simulator plot showing the 1.5 D representation of the linearly approximable feature being monitored superimposed with the optimal solution computed. For the instance shown in sub-figure (a), UGV was not used in the optimal solution. The symbols have the same meaning as defined in figure 3.

(a) (b)

Fig. 6: (a) Robots and (b) Linearly-approximable feature within the terrain used for the experiment.

reasonable sized instances of the problem that are encountered in typical missions. We have also developed a branch-and-cut based approach to address the exponential size of the formulation. Extensive simulations are performed to corroborate the performance of the approach. We have also conducted preliminary field trials to experimentally validate the proposed algorithm. As extensions to this work, the 2.5D version of the problem may be addressed. There is also a need to develop approximation algorithms that have bounds on the quality of the solution.

REFERENCES

[1] R. R. Murphy, *Disaster robotics*. MIT press, 2014.

[2] M. Dunbabin and L. Marques, "Robots for environmental monitoring: Significant advancements and applications," *IEEE Robotics and Automation Magazine*, vol. 19, no. 1, pp. 24 –39, Mar 2012.

[3] 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.

[4] P. Maini, G. Gupta, P. Totekar, and S. P. B., "Visibility-based monitoring of a path using a heterogeneous robot team," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2018.

[5] S. Alamdari, E. Fata, and S. L. Smith, "Persistent monitoring in discrete environments: Minimizing the maximum weighted latency between observations," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 138–154, 2014.

[6] W. Chin and S. Ntafos, "Optimum watchman routes," in *Proceedings of the Second Annual Symposium on Computational Geometry*, ser. SCG '86, 1986, pp. 24–33.

[7] A. Efrat, M. Nikkilä, and V. Polishchuk, "Sweeping a terrain by collaborative aerial vehicles," in *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ser. SIGSPATIAL'13, 2013, pp. 4–13.

[8] P. Tokekar and V. Kumar, "Visibility-based persistent monitoring with robot teams," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 3387–3394.

[9] S. Carlsson, H. Jonsson, and B. J. Nilsson, "Finding the shortest watchman route in a simple polygon," *Discrete & Computational Geometry*, vol. 22, no. 3, pp. 377–402, Oct 1999.

[10] S. Khuller, A. Malekian, and J. Mestre, "To fill or not to fill: The gas station problem," *ACM Transactions on Algorithms (TALG)*, vol. 7, no. 3, p. 36, 2011.

[11] K. Sundar and S. Rathinam, "Algorithms for routing an unmanned aerial vehicle in the presence of refueling depots," *Automation Science and Engineering, IEEE Transactions on*, vol. 11, no. 1, pp. 287–294, 2014.

[12] ——, "Generalized multiple depot traveling salesmen problempolyhedral study and exact algorithm," *Computers & Operations Research*, vol. 70, pp. 39–55, 2016.

[13] P. Maini and P. B. Sujit, "On cooperation between a fuel constrained uav and a refueling ugv for large scale mapping applications," in *International Conference on Unmanned Aircraft Systems*, June 2015, pp. 1370–1377.

[14] P. Maini, K. Sundar, S. Rathinam, and P. B. Sujit, "Cooperative planning for fuel-constrained aerial vehicles and ground-based refueling vehicles for large-scale coverage," *CoRR*, vol. abs/1805.04417, 2018.

[15] S. Carlsson and B. J. Nilsson, "Computing vision points in polygons," *Algorithmica*, vol. 24, pp. 50–75, 1999.

[16] M. Grötschel, M. W. Padberg *et al.*, "Polyhedral theory," *The traveling salesman problem*, pp. 251–305, 1985.