2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 27 - October 1, 2021. Prague, Czech Republic

2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-1714-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IROS51168.2021.9636549

Planning for Aerial Robot Teams for Wide-Area Biometric and Phenotypic Data Collection

Shashwata Mandal[1], Tianshuang Gao[1] and Sourabh Bhattacharya[1,2]

***Abstract*— This work presents an efficient and implementable solution to the problem of joint task allocation and path planning in a multi-UAV platform. The sensing requirement associated with the task gives rise to an uncanny variant of the traditional vehicle routing problem with coverage/sensing constraints. As is the case in several multi-robot path-planning problems, our problem reduces to an mTSP problem. In order to tame the computational challenges associated with the problem, we propose a hierarchical solution that decouples the vehicle routing problem from the target allocation problem. As a tangible solution to the allocation problem, we use a clustering-based technique that incorporates temporal uncertainty in the cardinality and position of the robots. Finally, we implement the proposed techniques on our multi-quadcopter platforms.**

I. INTRODUCTION

In the past decade, there has been a widespread deployment of unmanned aerial vehicles (UAVs) for surveillance in civilian as well as military applications [1] that require exploration of interest points [2] for situational awareness. Although UAVs have been effective in wide-area surveillance operations involving tracking of salient entities, their capabilities are encumbered by the amount of onboard power; fuelling the need for energy-efficient trajectory planning. This accompanies several challenges due to the close coupling between the low-level continuous-time optimal control problem involving the dynamics of the individual UAV and the discrete-time combinatorial optimization problems that arise at the team level. In this work, we address a joint allocation and path planning problem that arises when a team of UAVs is deployed to collect biometric/phenotypic data.

In general, vehicle routing problems reduce to some variant of the famous Travelling Salesman Problem (TSP) [3], [4], [5], [6]. In this paper, we deal with a TSP variant called as mTSP where m salesmen are initially located at a depot. Given a set of cities, and a cost metric, the goal is to calculate a set of routes for the m salesmen so that the total sum of the cost of the m routes is minimized. Thought mTSP is an NP-hard problem [7], approximation techniques for TSP (e.g. [3], [4], [5], [6], [8], [9] exists since mTSP can be reduced to a standard TSP.

Minimizing the energy consumption of a network to increase its average lifetime is a well-studied problem [10]. In robotic networks, this often translates to solving an optimal control problem for minimizing a metric related to the energy expended by the robot, for example, distance traveled [11], time required for task completion [12], wheel rotation [13], to name a few. In the past, researchers have studied energy-optimal trajectory generation for quadcopters [14] [15] [16] [17]. For time-optimal motion planning of quadcopters, a commonly used analytical technique is the Pontryagin Minimum Principle [18] [19] [20]. This approach provides the motion primitives of the optimal trajectory. The challenge here is to use the motion primitives for synthesis of the complete optimal trajectory between an initial and a final state. Non-linear programming-based approaches [21] [22] [23] first generate the control points, and then parameterize the generated path in time such that the dynamic constraints are enforced. Moreover, for generating smooth flight path, strategies on minimum snap trajectory planning have been proposed in [24] [25] [26].

The contributions of this paper are as follows: (i) We present a methodology to develop trajectory generation algorithms that can be implemented on aerial vehicles deployed in applications related to surveillance based data collection. (ii) We present a clustering approach to the UAV-target allocation problem that is scalable in the number of targets and UAVs (iii) The allocation approach presented in this paper can incorporate temporal uncertainties in the number and position of targets. (iv) The proposed techniques are implementable on a multi-UAV platform as demonstrated by the experiments.

The paper is organized as follows. Section II presents the problem formulation. Section III presents the solution approach. Section IV proposes a clustering-based technique for target allocation. Section V presents efficient trajectory generation technique for the quadcopter that take into account the sensing requirements and vehicle dynamics. Section VI presents the experimental set up and associated results. Section VII presents the conclusion along with the future work.

II. PROBLEM FORMULATION

We consider a problem in which a team of robots equipped with a vision sensor surveils a region with the objective of acquiring a 360° view of each target. Since we assume that the robots are fewer in number compared to the targets, each robot has to visit multiple targets.

Let $T = \{t_1, \ldots, t_n\}$ denote the team of n targets and $R = \{r_1, \ldots, r_m\}$ denote the team of m robots. We assume that all the robots start at a point called a *depot* denoted as s. Let $d_s : T \rightarrow \mathbb{R}$ denote the distance traveled by the robot

*This work was in part supported by USDA/NIFA National Robotics Initiative under Grant 2017-67021-25965 and NSF IIS award 1816343.

[1]Department of Computer Science, Iowa State University, Ames, IA 50010, USA, [2]Department of Mechanical Engineering, Iowa State University, Ames, IA 50010, USA smandal@iastate.edu tsgao@iastate.edu sbhattac@iastate.edu

between the depot and the target. Let $d : T \times T \rightarrow \mathbb{R}$ denote the distance traveled by the robot between targets. A robot r_i is tasked with a sequence π_i of k_i targets K_i such that $\bigcup_{i \in R} K_i = T$. Let Π be the ordered set of permutations such that $\Pi = \{\pi_i | i \in R\}$. In order to capture a 360° scan of the target, the robot rotates a full circle around it before visiting the next target. The radius of the circle is determined by two factors - 1) the safe distance from the target, 2) distance between the camera and the target; required to capture a high quality image. Therefore, the total distance traveled by r_i, denoted as C_i, to follow a sequence of targets in π_i is :

$$C_i(\pi_i) = d_s(\pi_i(1)) + \sum_{j=1}^{k_i - 1} d(\pi_i(j), \pi_i(j+1)) + \sum_{j=1}^{k_i} h(\pi_i(j)),$$

where h denotes the distance traveled by the robot around a target to acquire a 360° scan. The total cost incurred by the team of robots associated with an ordered set of permutation Π is given as follows:

$$C(\Pi) = \sum_{i \in R} C_i(\pi_i) \tag{1}$$

In order to minimize the energy spent in the surveillance task, we pose the joint allocation and path planning problem for the robots as an optimization problem of minimizing the total distance covered by the team of robots. In other words, the objective is to find the permutation Π^* defined as follows:

$$\Pi^* = \arg\min_{\Pi} C(\Pi) \tag{2}$$

In this work, we assume that the robots are quadcopters. For a target modeled as a point and robot modeled as a holonomic vehicle, the problem in (2) is an mTSP problem. Our problem is a variant of the mTSP problem with additional constraints of target loitering and vehicle dynamics.

III. Hierarchical Planning

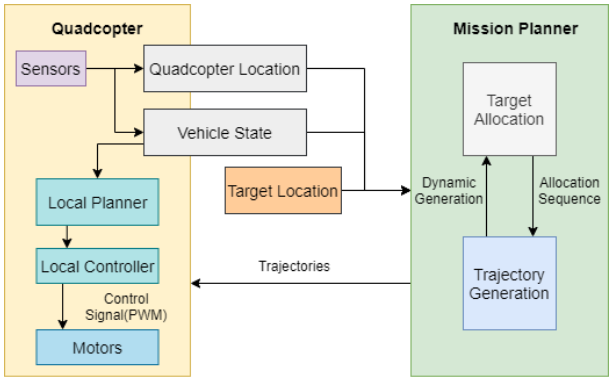


Fig. 1: Planner

Figure 1 shows our proposed solution. The overall planner can be divided into two major components:

1) Mission Planning: This module is tasked with the generation of the route along with the supported trajectory for the system using the current location of the targets as input. *Target Allocation* processes the incoming data and generates the allocation and visiting sequence(interchangeably called path in this paper) for each quadcopter periodically. *Trajectory Generation* receives the output from the target allocation module and generates the trajectories based on the dynamic model of the quadcopter which is fed back to the target allocation module to re-allocate the targets for the incoming/moving targets in a dynamic scenario. This is a part of the ground station. The majority of our work is focused on this mission planning algorithm itself.
2) Quadcopter : This module serves as an actuator for the algorithm and is a part of every quadcopter. The *Quadcopter* uses the generated trajectory as its flight plan. *Local planner* combines the trajectory and the current state of the quadcopter in order to compute the desired motion. Finally, *Local controller* sends the control signal (PWM) directly to the motors to follow the trajectory and complete the task. The hardware components include:
   a) Sensors : They are responsible for keeping track of the current system states
   b) Quadcopter Location : This refers to the GPS module on-board which keeps track of the physical location of the quadcopter

The hierarchical nature of the planner stems from the two-step trajectory generation strategy with the first stage being path generation followed by its upgradation to a minimum time trajectory at the end of the two stages. As described this can be broken down into separate modules which together form the Mission Planning Algorithm:

1) Clustering-Based Target Allocation - Generates a path connecting a subset of the targets for each of the quadcopters such that all the targets are covered without overlap using the location of targets as input.
2) Trajectory Generation - Converts the path (or order of visitation) for the targets generated by the previous module and converts it into a trajectory. This module is responsible for generating the minimum time trajectory between all pairs of targets in a given path. Once a trajectory is generated for all the paths provided by the previous module, this can be send to the quadcopter to begin its surveillance.

The next two sections describe in detail the working of each of the aforementioned modules.

IV. Clustering-based Target Allocation

In this section, we address the allocation problem where a team of quadcopters(salesmen) are allocated to a pack of targets(cities). The original problem as posed in (2) is an mTSP problem which is well-known to be NP-hard [7]. Current known solutions to the mTSP problem involve heuristic [27] and approximation [28]. Since the space in our problem is metrizable, we can reduce it to a standard Δ-TSP problem. In Δ-TSP, there are n cities with a distance function in the form of $d : [n] \times [n] \rightarrow \mathbb{R}$ where d is set in a metric space. The objective is to find a permutation π which minimizes the

total distance $D = \sum_{j=1}^{n-1} d(\pi_i(j), \pi_i(j+1) + d(\pi(1), \pi(n))$. Therefore, problem (2) is equivalent to solving a multi-variate version of the standard Δ-TSP.

Even though Δ-TSP is known to be NP-Hard [29], the Christofides' algorithm provides a $\frac{3}{2}$ approximation algorithm for Δ-TSP [30]. Leveraging on Christofides' algorithm, Algorithm 2 presents a hierarchical approach to the allocation problem using a clustering technique. The crucial part of the solution is to first partition the cities into m clusters using a K-means clustering algorithm [31] (where m is the number of quadcopters) and then compute a tour for each of these m clusters using Christofides algorithm. In Line 2, the targets are divided into m clusters. It may be noted that the K-means algorithm may be replaced by any other clustering algorithm. An advantage of this approach is its flexibility in accommodating additional targets in the environment without recomputing the clusters. Line 3 is an optional optimization step for decreasing the number of targets for surveillance. The central idea here is to use hierarchical clustering [32] to represent targets which are really close to each other as a single target. In Lines 5-8, we run Christofides Algorithm on each of the above mentioned clusters. Finally, once the tours are constructed for each of the clusters, the tours are forwarded to the trajectory generation algorithm. The distribution using K-means does not consider equal distribution of the targets to all the quadcopters since K-means enforces spatially close targets to cluster together. We allow this behavior to persist since if equal distribution was somehow enforced, it could result in a target that is quite distant to a quadcopter to be assigned to it. This would increase the overall path cost.

Algorithm 1 Assigning n targets to m quadcopters

Input: m quadcopters Q, depot location s, n starting targets with their locations T, distance functions d_s and d

Output: Targets assignment A - m ordered sets with each set representing the order of targets visited by that quadcopter

1: **function** TARGETASSIGNMENT(Q, T, s, d, d_s)
2: $A_{K-means} \leftarrow$ K-MEANS(m, T)
3: $A'_{K-means} \leftarrow$ TRUNCATE-TARGETS($A_{K-means}$)
4: **for** each set $q \in T'_{K-means}$ **do**
5: Construct graph G_q from T using q and d
6: $A_q \leftarrow$ CHRISTOFIDES(G_q) $\quad /* A_q$ contains the ordered set of targets visited in the TSP tour $*/$
7: Find the closest point in A_q from s and set it as the beginning of the tour
8: Append A_q to A
9: **end for**
10: **return** A
11: **end function**

Since the trajectories of the quadcopter are obtained from numerical optimal control, it is challenging to obtain a theoretical gap between the cost incurred by the team when the quadcopters are approximated by a holonomic vehicle. Figure 3 shows the average, maximum and minimum ratio

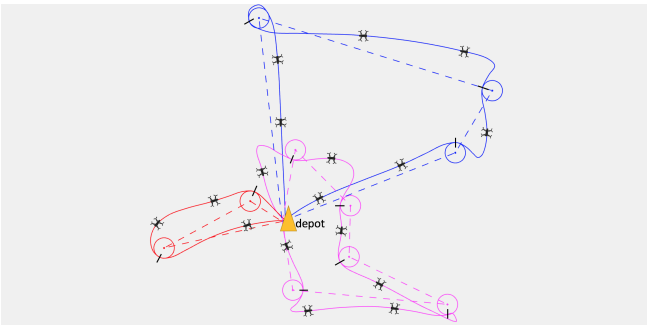


Fig. 2: Figure shows the trajectory (solid lines) of the quadcopters based on the numerical optical control proposed in Section V, and the (dashed lines) trajectory generated by Algorithm 2.

between the minimum time required by a holonomic vehicle and a quadcopter for the transition phase for different initial position of the vehicles. From the simulation results, we can conclude that the distance traveled by quadcopters is at most 3 times that of a holonomic vehicle. For an mTSP problem, [28] shows that if the number of paths is m then the overall approximation factor can be further reduced to $\frac{3}{2} - \frac{1}{m}$. Combining the $\frac{3}{2} - \frac{1}{m}$ approximate bound of christofides with the empirically obtained bound of 3 from simulation results leads to an overall $4.5 - \frac{3}{m}$ approximation factor for the cost incurred by the quadcopters relative to the optimal route for holonomic vehicles using the allocation technique proposed in Algorithm 2.

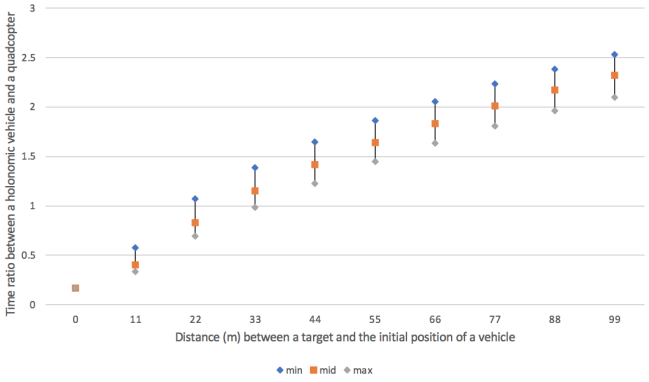


Fig. 3: Avg-Max-Min time radio between holonomic vehicle without motion constraint and our result.

A. Incorporating Variable and Mobile Targets

Since the overall motivation is to build a deployable multi-quadcopter platform for real-time surveillance, the system should be able to adapt to uncertainties in the number and location of the targets. The technique proposed in the previous section for static targets can be considered as a snapshot of the dynamic scenario regarding the state of the targets. In this section, we extend the static algorithm to address situations in which new targets may appear or old targets may disappear from the surveillance region.

In Algorithm 2, the quadcopters return to the depot after completing their original tours. Algorithm 2 doesn't allow

the quadcopters to return to the depot but instead initiates a new tour (Line 7-9) of the updated cluster (Line 4). The algorithm includes a target absorption phase where a new target is absorbed by the existing clusters based on their distance from their cluster centers (Line 4). An additional step includes re-evaluating the TSP circuit for each cluster once the previous iteration of drone surveillance is finished (Line 7-9). Once the TSP circuits are evaluated the drones need to be send on these circuits to survey the targets. This is done by the TRIP procedure (Line 10). The method allows us to absorb new targets to the original clusters overtime. However, a downside to this is once too many new targets are absorbed the cluster might differ significantly in comparison with a freshly calculated K-means cluster leading to a drop in efficiency overtime. This can be rectified by reevaluating the cluster after r_t iterations. The reevaluation time r_t sets the number of iterations after which the clusters will be reevaluated. For our simulation we keep $r_t = \infty$. i.e. it never reevaluates. One more assumption we make is the surveillance targets need to be constantly surveyed in a loop overtime i.e. if a target is surveyed at time t, it must be resurveyed at sometime $t + \Delta$.

Algorithm 2 Dynamic target assignment to n quadcopters

Input: m quadcopters Q, depot location s, n targets along with their locations T, distance functions d_s and d, reevaluation timer r_t

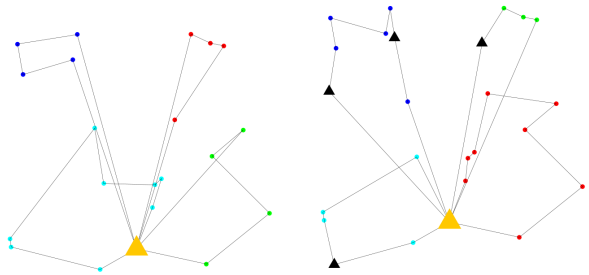
1: **function** DYNAMIC ASSIGNMENT(Q, T, s, d, d_s)
2: $A_{K-means} \leftarrow$ K-MEANS(M) /*Call K-means and store assignment in $A_{K-means}$*/
3: Move all quads to the nearest target in their clusters
4: Asynchronously assign new targets to nearest cluster
5: $A'_{K-means} \leftarrow$ TRUNCATE-TARGETS($A_{K-means}$)
6: **for** each set $q \in A'_{K-means}$ **do** /*Run in parallel*/
7: Construct graph G_q from T using q and d
8: $A_q \leftarrow$ CHRISTOFIDES(G_q) /*A_q contains the ordered set of targets visited in the TSP tour*/
9: Find the closest point in A_q from s and set it as the beginning of the tour if this is the first iteration
10: Execute TRIP(A_q) /*This signifies executing the trip assigned to quadcopter q asynchronously*/
11: **end for**
12: Reset after r_t iterations.
13: **end function**

Given a scenario where a set of n original targets exist and r new targets are introduced, there are two ways of solving the surveillance problem. The first method is to use Algorithm 1 to finish surveillance of the original targets. After the new targets are introduced, complete a fresh execution of Algorithm 1 on the updated set of targets. The second method is to use Algorithm 2 to complete surveillance of the original targets. After the new targets are introduced, they are absorbed by the existing clusters, and Algorithm 2 completes surveillance of the new clusters in its second iteration.

Figure 5 demonstrates the procedures when new targets are introduced in the environment. We can see that Procedure 1 relies on completing the tour and returning to the depot where as Procedure 2 completes the tour and stays on the final target waiting for the new iteration. This creates a difference in terms of the total path cost.

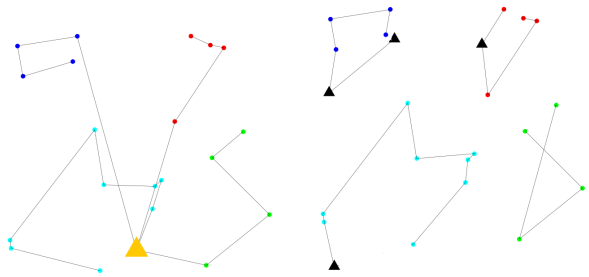
r (number of target)	32	64	100	150
ρ (ratio of path lengths)	1.22	1.20	1.19	1.18

Fig. 4: Comparing the ratio (ρ) of the path lengths in Procedure 1 and Procedure 2 when r new targets are added, where $n = 100$, $m = 8$ and number of simulations = 30000



(a) Algorithm 1 - Run 1

(b) Algorithm 1 - Run 2 with new targets



(c) Algorithm 2 - Iteration 1

(d) Algorithm 2 - Iteration 2 with new targets

Fig. 5: Figure shows the two procedures applied to the dynamic scenario where new targets (represented by black triangles) arrive. The orange triangle represents the depot.

Figure 4 shows the results for the simulation where new targets are introduced after the old targets have been processed. The simulation is initiated with 100 targets (n) and 8 quadcopters (m). Once the first round of processing is completed, r new targets are introduced. After the second round of processing, the path lengths generated from the two procedures are compared and analysed. The ratio ($\rho = \frac{PathLength_{Procedure1}}{PathLength_{Procedure2}}$) is the metric for the comparison. We see the results for the above mentioned simulation for $r = 32, 64, 150$. We see that the path lengths for Procedure 2 is less than that for Procedure 1 even after 150 new targets are introduced. This clearly demonstrates how Algorithm 2 outperforms Algorithm 1 at handling dynamic scenarios.

The simulations in Figure 5 were created in java using the

StdDraw library to generate random points over a space of $X : [-1000, 1000] Y : [-1000, 1000]$. Needless to say, the figure shows only 10 targets but this can be scalable to 100 or more targets. It was infact test with upto 150 targets.

Once the TSP circuits for each of the quadcopter have been generated, they can be sent to the next module for generating the minimum-time trajectory.

V. Trajectory Generation

In this section, we describe the different phases of the quadcopter's proposed trajectory.

1) Loitering Phase: In the loitering phase, the quadcopter scans a target while traversing a circular trajectory at its maximum speed at a constant.

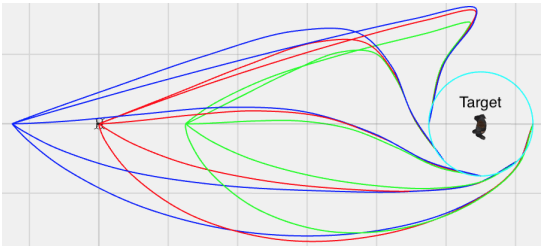


Fig. 6: Minimum-time trajectories for three different initial states of the quadcopter for five equi-spaced terminal points on the semicircle. The target circle is shown in light blue color.

2) Transition Phase: In this phase, the quadcopter leaves the circle around one target and arrives on the circle around the next target. The state it leaves a circle is where it entered the circle earlier. The minimum-time control problem of the quadcopter can be formulated as the following Bolza problem [33]:

$$J = \int_0^{t_f} 1 \quad dt, \tag{3}$$

where t_f denotes the time at which the quadcopter reaches the circle. Due to the non-linearity in the dynamics of the quadcopter, (3) is solved using Optimtraj library [34]. Since the non-linear optimization is a two-point boundary value problem, the final circle is discretized to determine the values of the state to enter the circle. At the final state, the direction of rotation of the quadcopter around the target can be clockwise or anti-clockwise. This leads to two optimization problems that need to be solved for a given final state. Fig. 6 shows the minimum-time trajectory for three different initial distance from the circle for five equispaced terminal points on the semicircle. Algorithm 3 shows the process to generate the minimum-time trajectory.

VI. Experiments

The strategy proposed in this paper is implemented on a multi-UAV testbed developed in our lab called *Cy-Eye* [35]. Figure 7 shows the communication architecture.

We deployed the quadcopters to encircle 8 targets in an open space (50x50m). The ground control allocates the

Algorithm 3 Trajectory Generation between Targets ($s \rightarrow t$)

Input: Point p where a quadcopter leaves target s, Target t, radius r

Output: Trajectory T

```
function TrajectoryGenerator(p, t)
    Discretize points on circles t
    Sample entry points S on circle t
    for each entry points e ∈ S do
        Calculate the time-optimal trajectory between p and e using Non-linear optimizer
    end for
    Find the trajectory T that has the shortest time
    return T
end function
```

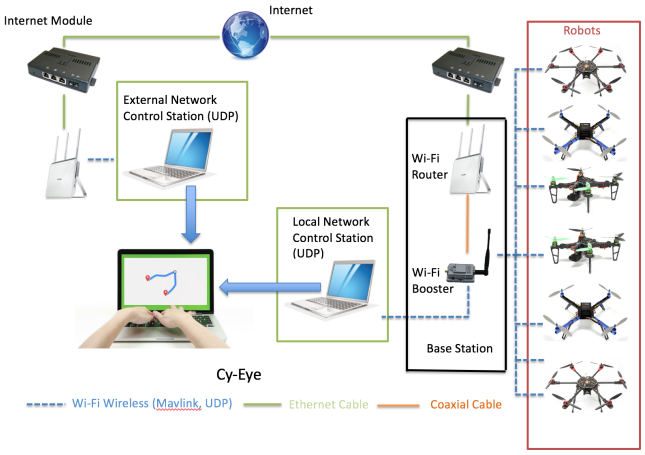


Fig. 7: Communication architecture of Cy-Eye.

targets for each quadcopter by using Algorithm 1. The trajectories generated by Algorithm 3 are sent to each quadcopter through UDP/MAVLink protocol. The parameters in the dynamic model (Inertial, propulsion, air density, thrust, etc.) are measured based on our custom-build quadcopter (450mm). In the experiment, we set the flying altitude between 1m and 2m. Once the trajectories are received, the quadcopters start their task automatically by following the trajectories.

Figure 8 shows the path traversed by three quadcopters using our proposed algorithm. The actual flight paths (shown with dashed line) are longer than the trajectories generated by Algorithm 1. In the experiment, the three drones take 120 seconds to complete the task. Comparing to the flying time (113s) from the simulation, a longer path leads to a longer flying time, the result shows that the system could successfully complete the task of encircling the targets. A video accompanying the submission provides details regarding the experiments.

VII. Conclusion and Future Work

In this work, we present a strategy to generate trajectories that can be implemented on aerial robots deployed in a surveillance application. The problem is well-known to be NP-hard. The hierarchical approach that we proposed divides this problem into 2 sub-problems (Targets allocation and trajectories generation). We present a clustering approach

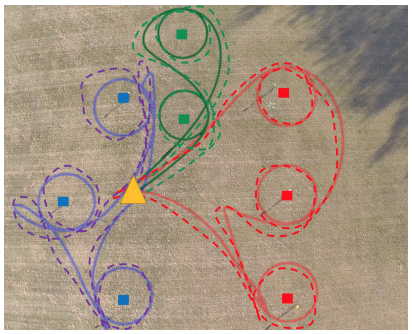


Fig. 8: Figure shows the paths generated from Algorithm 3 (solid), and the actual paths (dashed) traversed by the quadcopters during the experiment.

to the allocation problem that is scalable in the number of targets and robots. A non-linear programming-based trajectories generation approach is presented. We investigate the proposed strategies through extensive simulation. Finally, the same strategy was applied on a multi-UAV platform in the outdoor environment.

As a future research direction, we plan to investigate the performance of the system for several applications. Of particular interest, are applications related to intruder identification and livestock phenotyping. An ongoing research direction is the detection of face masks for preventing respiratory infections diseases in humans. We are specifically interested in implementing real-time machine learning algorithms in the quadcopter perception loop.

REFERENCES

[1] A. Ollero and I. Maza, *Multiple heterogeneous unmanned aerial vehicles*. Springer, 2007, vol. 37.

[2] E. Bahceci, O. Soysal, and E. Sahin, "A review: Pattern formation and adaptation in multi-robot systems," *Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-03-43*, 2003.

[3] B. Golden, L. Levy, and R. Dahl, "Two generalizations of the traveling salesman problem," *Omega*, vol. 9, no. 4, pp. 439–441, 1981.

[4] M. M. Flood, "The traveling-salesman problem," *Operations research*, vol. 4, no. 1, pp. 61–75, 1956.

[5] G. Laporte, "The traveling salesman problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, no. 2, pp. 231–247, 1992.

[6] G. Gutin and A. P. Punnen, *The traveling salesman problem and its variations*. Springer Science & Business Media, 2006, vol. 12.

[7] A. Király and J. Abonyi, "A novel approach to solve multiple traveling salesmen problem by genetic algorithm," in *Computational Intelligence in Engineering*. Springer, 2010, pp. 141–151.

[8] S. Gorenstein, "Printing press scheduling for multi-edition periodicals," *Management Science*, vol. 16, no. 6, pp. B–373, 1970.

[9] M. Bellmore and S. Hong, "Transformation of multisalesman problem to the standard traveling salesman problem," *Journal of the ACM (JACM)*, vol. 21, no. 3, pp. 500–504, 1974.

[10] V. Rodoplu and T. H. Meng, "Minimum energy mobile wireless networks," *IEEE Journal on selected areas in communications*, vol. 17, no. 8, pp. 1333–1344, 1999.

[11] E. F. Mohamed, K. El-Metwally, and A. Hanafy, "An improved tangent bug method integrated with artificial potential field for multi-robot path planning," in *2011 International Symposium on Innovations in Intelligent Systems and Applications*. IEEE, 2011, pp. 555–559.

[12] K. Jiang, L. D. Seneviratne, and S. Earles, "Time-optimal smooth-path motion planning for a mobile robot with kinematic constraints," *Robotica*, vol. 15, no. 5, pp. 547–553, 1997.

[13] H. Chitsaz, S. M. LaValle, D. J. Balkcom, and M. T. Mason, "Minimum wheel-rotation paths for differential-drive mobile robots," *The International Journal of Robotics Research*, vol. 28, no. 1, pp. 66–80, 2009.

[14] I. D. Cowling, O. A. Yakimenko, J. F. Whidborne, and A. K. Cooke, "Direct method based control system for an autonomous quadrotor," *Journal of Intelligent & Robotic Systems*, vol. 60, no. 2, pp. 285–316, 2010.

[15] A. Chamseddine, Y. Zhang, C. A. Rabbath, C. Join, and D. Theilliol, "Flatness-based trajectory planning/replanning for a quadrotor unmanned aerial vehicle," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 48, no. 4, pp. 2832–2848, 2012.

[16] S. Taamallah, X. Bombois, and P. M. Van den Hof, "Trajectory planning and trajectory tracking for a small-scale helicopter in autorotation," *Control Engineering Practice*, vol. 58, pp. 88–106, 2017.

[17] F. Morbidi, R. Cano, and D. Lara, "Minimum-energy path generation for a quadrotor uav," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1492–1498.

[18] R. Ritz, M. Hehn, S. Lupashin, and R. D'Andrea, "Quadrocopter performance benchmarking using optimal control," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 5179–5186.

[19] M. Hehn and R. D'Andrea, "Quadrocopter trajectory generation and control," *IFAC proceedings Volumes*, vol. 44, no. 1, pp. 1485–1491, 2011.

[20] M. Hehn, R. Ritz, and R. D'Andrea, "Performance benchmarking of quadrotor systems using time-optimal control," *Autonomous Robots*, vol. 33, no. 1-2, pp. 69–88, 2012.

[21] L.-C. Lai, C.-C. Yang, and C.-J. Wu, "Time-optimal control of a hovering quad-rotor helicopter," *Journal of Intelligent and Robotic Systems*, vol. 45, no. 2, pp. 115–135, 2006.

[22] Y. Bouktir, M. Haddad, and T. Chettibi, "Trajectory planning for a quadrotor helicopter," in *2008 16th mediterranean conference on control and automation*. Ieee, 2008, pp. 1258–1263.

[23] E. Kahale, P. Castillo, and Y. Bestaoui, "Minimum time reference trajectory generation for an autonomous quadrotor," in *2014 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2014, pp. 126–133.

[24] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE international conference on robotics and automation*. IEEE, 2011, pp. 2520–2525.

[25] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012.

[26] J. Yu, Z. Cai, and Y. Wang, "Optimal trajectory generation of a quadrotor based on the differential flatness," in *2016 Chinese Control and Decision Conference (CCDC)*. IEEE, 2016, pp. 678–683.

[27] Pan Junjie and Wang Dingwei, "An ant colony optimization algorithm for multiple travelling salesman problem," in *First International Conference on Innovative Computing, Information and Control - Volume I (ICICIC'06)*, vol. 1, 2006, pp. 210–213.

[28] G. N. Frederickson, M. S. Hecht, and C. E. Kim, "Approximation algorithms for some routing problems," in *17th Annual Symposium on Foundations of Computer Science (sfcs 1976)*, 1976, pp. 216–227.

[29] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms, Third Edition*, 3rd ed. The MIT Press, 2009.

[30] N. Christofides, "Worst-case analysis of a new heuristic for the travelling salesman problem," Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group, Tech. Rep., 1976.

[31] R. Nallusamy, K. Duraiswamy, R. Dhanalaksmi, and P. Parthiban, "Optimization of non-linear multiple traveling salesman problem using k-means clustering, shrink wrap algorithm and meta-heuristics," *International Journal of Nonlinear Science*, vol. 9, no. 2, pp. 171–177, 2010.

[32] D. Defays, "An efficient algorithm for a complete link method," *Comput. J.*, vol. 20, pp. 364–366, 1977.

[33] H. H. Goldstine, *A History of the Calculus of Variations from the 17th through the 19th Century*. Springer Science & Business Media, 2012, vol. 5.

[34] M. Kelly, "Optimtraj – trajectory optimization library." [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/54386-optimtraj-trajectory-optimization-library

[35] T. Gao, H. Emadi, H. Saha, J. Zhang, A. Lofquist, A. Singh, B. Ganapathysubramanian, S. Sarkar, A. K. Singh, and S. Bhattacharya, "A novel multirobot system for plant phenotyping," *Robotics*, vol. 7, no. 4, p. 61, 2018.