Social Drone Sharing to Increase the UAV Patrolling Autonomy in Emergency Scenarios.

Luca Morando, Carmine Tommaso Recchiuto and Antonio Sgorbissa

Abstract—Unmanned Aerial Vehicles (UAVs) popularity is increased in recent years, and the domain of application of this new technology is continuously expanding. However, although UAVs may be extremely useful in monitoring contexts, the operational aspects of drone patrolling services have not yet been extensively studied. Specifically, patrolling and inspecting with UAVs different targets distributed over a large area is still an open problem, due to battery constraints and other practical limitations. In this work, we propose a deterministic algorithm for patrolling large areas in a pre- or post-critical event scenario. The autonomy range of UAVs is extended with the concept of Social Drone Sharing: citizens may offer their availability to take care of the UAV if it lands in their private area, being thus strictly involved in the monitoring process. The proposed approach aims at finding optimal routes in this context, minimizing the patrolling time and respecting the battery constraints. Simulation experiments have been conducted, giving some insights about the performance of the proposed method.

I. INTRODUCTION

Multirotor drone technology is available on a large scale on market and widely utilized in research. Indeed multirotors are endowed with several specific features such as vertical take-off and landing, the ability to hover freely in the air, the reduced size and the possibility to be programmed, that make them very interesting in different scenarios. Thanks to all these characteristics, drones are now used in mapping [1], in agriculture [17], in inspection sites [2] and soon they may be also used in delivery. Emergency scenarios are another domain where UAVs are becoming increasingly important [21]. Indeed there are various examples in literature, starting from the nuclear disaster of Fukushima [3] to the pandemic disease caused from the virus Sars-Cov-2, where UAVs have been successfully employed: in particular, drones have been utilized to control the body temperature of a multitude of people in the street during quarantine periods [4]. Despite their effectiveness, a significant limit to the use of this technology lies in the absence of true autonomy. This problem is twofold:

- autonomous navigation behavior, i.e. the capability to autonomously coordinate a team of UAVs to perform a patrolling operations, is missing, thus requiring expert pilots also in the simpler operations.
- energetic autonomy, i.e., the capability to cover long distances without the need to be recharged, is limited to a time of flight of 15-30 minutes, thus resulting in a reduced operative area.

Our proposal lies in the context of pre-operative or post-operative intervention in emergency scenarios, where rescue teams or the civil protection need to periodically monitor or survey different critical targets located in various places taking as little time as possible. These targets could be, for instance, fields or woods whose dryness shall be periodically assessed to prevent fires, soil slopes whose stability shall be monitored before or after floods, critical infrastructure (hospitals, schools, roads, bridges, etc.) whose risk shall be evaluated before or after earthquakes. In most cases the maximum drone flight time, generally defined by the drone technical specification, is not sufficient to visits all targets. For these reasons, the proposed approach aims at involving citizens in the monitoring process, by introducing the new concepts of Social Charging Stations (CS) and Social Drone Sharing (SDS).

More in detail, we foresee a software architecture composed by a platform that permits to manage the drones, to communicate and coordinate their flight plans depending on the type of scenario (pre-event, during or after). Civil authorities can interface with the platform to change the targets to survey depending on the need, and also visualize the CS available in the operative area. Resident volunteers give their availability through a mobile application, which alerts the user if their help is required to prepare the landing spot and to recharge the batteries if the drone lands on their property. The purpose is that residents volunteers shall be actively involved in risk assessment and management areas, which may be particular important especially in areas with low population density where a community-based approach is repaid by a greater safety. The general platform architecture is described in the proposed scheme in Figure 1.

Since resident volunteers are involved, which raises a number of logistics and legal issues, it is important to minimize the number of volunteers / charging stations utilized and at the same time maximize the inspection time in a monitored area. To this aim, this work describes an approach for finding optimal routes for autonomously flying UAVs in patrolling contexts, taking into account the new concepts of Social Drone Sharing.

A. Problem Statement

Drones applications in emergency scenario are even more diffused, because they permit to fly over a dangerous or compromised area, making a possible inspection of otherwise unreachable targets. Actually these operations are conducted
Fig. 1: Operative Platform architecture description

by expert pilots in rescue teams, who define an initial location where the drone shall take-off and land at the end of the operations. This technique reduces the operative area because the UAV has to stay inside the pilot Visual Line Of Sight (VLOS) and, when the battery level becomes critical, the drone must return to the same take-off area.

Our approach is based on the utilization of a single UAV, or a team of completely autonomous drones which are given by authorities to resident volunteers. The idea is that people give their availability in securing a recharging station spot in their property, such as a terrace or a garden, that a drone may use if it requires to be recharged. Volunteers can communicate their commitment to the service to the civil protection, which proceeds to collocate the social charging station (CS) in the citizens property. Finally the UAVs patrolling path can be computed knowing the geographic positions of the targets to be inspected and the CS. With this solution, a distribution of CS is obtained over the interested inspection area, resulting in increased energetic autonomy in terms of the distance that the UAV can travel between targets and/on an increased time which the UAV can spend in each Target.

The current approach is compared with our proposal in Figures 1.A and 1.B. The red squares are the targets while the blue circles represent the CS. The green circle, visible in both graphs, is the starting node, which has a different meaning in the two scenarios.

In the current approach, the starting node corresponds to the place where the drone pilot gets off his/her car, instructs the drone or the team of drones [19] to take-off, and then control it during all flight (typically, by keeping it within VLOS).

In the SDS approach, patrol rounds are performed periodically according to the scheduled flight plans: this means that the starting node simply corresponds to the Charging Station where the UAV finally landed when the previous tour ended, and from where it will take-off in the next tour after being recharged by the resident volunteer whose property has been selected for landing/take-off.

In the current approach, the drone is radio-controlled. In this case, the battery autonomy problem is very relevant because the drone needs every time to return where the operator is placed, to change or recharge the batteries.

In the SDS approach, CS are located in the correspondence of volunteers houses. In this case, drones are supposed to be fully autonomous and in this case all targets are reachable because (i) there is no need to cover all the targets with the range of the radio controller and (ii) the energetic autonomy range is increased.

In summary, the paper proposes an innovative solution for patrolling large areas by defining the new concept of SDS, and proposes an optimal and deterministic solution to solve the related problem. The approach, only based on two decision variables, permits to reduce the number of equations and the computational time of the optimization problem.

This work is organized as follow. Section 2 summarizes the relevant Literature. Section 3 formally describes the problem via Mixed Integer Linear Programming (MILP) formulation. The experimental results obtained in simulation via the MATLAB® optimization toolbox are exposed in Section 4. Finally, in Section 5 some conclusions and directives for future researches are drawn.

II. RELATED WORK

The Single Routing Drone (SRD) is a variation of the most common TSP problem. A recent review of the Traveling Salesman problem (TSP) can be found in [7].

Only in recent years a field of research based on the utilization of drones in different scenarios, from logistic to emergency, has started to grow. Several works are focused on the study and definition of new and more efficient algorithms for trajectory generation and obstacle avoidance, by limiting the analysis on short travel distances [9] [20]. Recently, researchers around the world have started to develop innovative solutions to stem the energy limitation of drones.
and other vehicles. The topic is receiving attention also by private companies: Apple has recently presented a software tool, integrated with the Apple Maps app and conceived for electric vehicles, which plans the fastest route between two points, also considering the possibility of stopping in recharging stations. \[22\]. However, coming back to the research world, the most common scenario is the truck-drone delivery problem, widely described in the survey proposed by \[10\]. In this scenario the drone is supposed to travel with a delivery truck, specially modified to charging the UAV. Another solution has been proposed by \[11\], where a Heuristic approach based on a validated mathematical definition it is used to find a mostly optimal solution which consists in a reduced delivery required time. The model proposed in \[11\] permits to optimize the truck delivery path taking into account that some customer locations can be visited also by the UAV that lands on the truck at the end of the operation.

In our knowledge, various work which has tried to find a solution to a similar problem are related to the utilization of terrestrial vehicle in a delivering scenario, generally solved by dynamic algorithms with a small number of targets and available charging stations. Very interesting is the work presented by \[14\], where a solution is proposed in the modeling of a delivery truck scenario. The truck must respect the delivery time constraint visiting all the targets before returning to the depot. The overall formulation is based on the traveled distances with the utilization of an integer decision variable to check if a station has been visited.

Differently, but always in a delivery context, \[12\] describes Parallel Drone Scheduling Traveling Salesman Problem PDSTSP, which is very similar to the context of our research. The PDSTSP describes a drone facility within a distribution center. In fact a sufficiently large number of drones can be utilized at the distribution center but the limited flight range creates a practical issue. The work proposed by \[13\] is directly derived by the model proposed in \[12\]. They proposed a scenario where drones have to visit different targets but only with a single charging station, where UAVs can land in case of insufficient autonomy.

Another interesting and more general work is defined by \[15\], also based in delivery with a truck scenario: a solution of the problem based on a greedy approach was proposed, in which the gas cost was incorporated to the objective function. The same approach is considered by \[18\] where the authors, after a detailed analysis of the energy consumption of the drones used in their experiment, proposed a solution to the drone routing problem via the Christofides Algorithm.

III. UAV SOCIAL CHARGING ROUTING PROBLEM (USCRP)

A. Notation

The USCRP problem follows the classical notation of the Vehicle Routing Problem: let \( G = (N \cup F, E) \) be a graph where \( N = \{1, 2, \ldots, i, \ldots, N\} \) is the set of the Target vertices from 1 to \( N \), while the set \( N0 = N \cup 0 \) contains also the starting node. \( F = \{0, N+1, N+2, \ldots, N+i, \ldots, N+F\} \) is the set of the \( F \) available charging stations in the graph. The starting node 0 can be considered also as a charging station. \( E = \{(i, j), i \neq j\} \) is the set of arcs connecting vertices \( i \) and \( j \). In USCRP, all nodes contained in the set of target nodes \( N0 \) must be visited exactly once. However, differently from the classical TSP problem, USCRP foresees a visit to a charging station node only when visiting that node is strictly necessary for completing the route, i.e. the autonomy of the drone is not enough to reach the remaining targets. As a result, some nodes in the set \( F \) could not be visited at all, while others may be visited multiple times in a given solution. Each arc \((i, j)\) presents three weights, defined by a non-negative distance \( d_{i,j} \), travel time \( t_{i,j} \) and an estimated battery consumption \( b_{i,j} \). Please notice that, in the following, we adopt a simplified model in which \( t_{i,j} \) and \( b_{i,j} \) can be derived from \( d_{i,j} \) through a simple scaling factor, and therefore they do not need to be stored. A detailed description about the calculation of the estimated battery consumption corresponding to each arc is given in the section below.

B. Energy Model

The energy model is derived from \[5\], where a UAV dynamical analysis is proposed. The authors propose a method to derive the motor angular speed \( \omega \), expressed in \( \text{rad sec}^{-1} \), starting from the quadrorotor equations and parameters given a constant horizontal or vertical translational velocity. From empirical results parameters, it is possible to fit the polynomial curve, visible in equation (1) which permits to obtain the motor Power consumption given the motor angular velocities (max 500 rad/s).

\[
P \approx 2.258 \times 10^{-7} w^3 + 3.866 \times 10^{-5} w^2 + 5.137 \times 10^{-3} w^1 + 2.616
\]

(1)

This equation model is evaluated taking into account the technical parameters of the DJI Mavic 2 Enterprise quadro-rotor \[6\]. Given the Maximum battery capacity (e.g., \( C_{\text{max}} = 213444 \) Wsec for the DJI Mavic 2\(^{1}\)) and a priori defined constant horizontal velocity \( v_h \), it is possible to evaluate the battery consumption normalized by the maximum battery capacity, as visible in equation (2), i.e., \( b_{i,j} \) can never be higher than 1 for the UAV to be able to move from \( i \) to \( j \).

\[
b_{i,j} = \frac{P_{t_{i,j}}}{C_{\text{max}}}
\]

(2)

where \( t_{i,j} = d_{i,j} v_h^{-1} \), is the time required to traverse the arc.

The UAV starts from the initial vertex 0 with the battery completely charged: if required to find a solution which visits all targets, it may decide to reach a charging station where its autonomy of the drone is not enough to reach the remaining targets. As a result, some nodes in the set \( F \) could not be visited at all, while others may be visited multiple times in a given solution. Each arc \((i, j)\) presents three weights, defined by a non-negative distance \( d_{i,j} \), travel time \( t_{i,j} \) and an estimated battery consumption \( b_{i,j} \). Please notice that, in the following, we adopt a simplified model in which \( t_{i,j} \) and \( b_{i,j} \) can be derived from \( d_{i,j} \) through a simple scaling factor, and therefore they do not need to be stored. A detailed description about the calculation of the estimated battery consumption corresponding to each arc is given in the section below.

\(^{1}\)https://www.dji.com/it/mavic-2
C. Mathematical model

The USCRP finds an optimal path solution, where it exists, (i) minimizing the traveled time plus the recharging time and (ii) respecting the battery constraints.

The following mathematical notation is proposed:

Objective parameters:
- \( t_{i,j} \): travel time on each arc \( i,j \) [sec].
- \( S_R, S_L \) and \( S_T \): respectively the time required to recharge the batteries, to land and to take off from the CS [sec].

Additional Graph Weights and variables:
- \( d_{i,j} \): length of each arc [Km].
- \( b_{i,j} \): expected battery consumption on each arc \([b_{min} < b_{i,j} < b_{max}]\), where \( b_{max} = 1 \) is the maximum battery value after recharging (normalized to 1) and \( b_{min} \) is the lower threshold of battery level.

Decision Variables:
- \( x_{i,j} \) remaining battery autonomy when the drone arrives at vertex \( j \) from vertex \( i \), with \( b_{min} < e_{i,j} < b_{max} \).

The mathematical formulation tracks the visiting order of the vertices and it is defined as a Mixed Integer Linear Programming formulation

\[
\min_{x_{i,j}} Z = \sum_{(i,j) \in E} t_{i,j} x_{i,j} + \sum_{(i,j) \in E, j \in F} (S_i + S_t + S_R) x_{i,j} \tag{4}
\]

subject to:
- \( \sum_{j \in N \cup F} x_{0,j} = 1 \tag{5} \)
- \( \sum_{i \in N \cup F} x_{i,0} = 1 \tag{6} \)
- \( \sum_{j \in N \cup F} x_{i,j} = 1, \forall i \in N \tag{7} \)
- \( \sum_{i \in N \cup F} x_{i,j} = \sum_{j \in N \cup F} x_{j,k}, \forall j \in N \cup F \tag{8} \)
- \( \sum_{i,j \in N \cup F} x_{i,j} \leq |S_v| - 1, \forall v, S_v \subset N \cup F \tag{9} \)

\( e_{i,j} \leq M - (M - b_{max} + b_{i,j}) x_{i,j} , \forall i \in N, \forall j \in N \cup F \tag{10} \)

\( -e_{i,j} \leq M - (M + b_{max} - b_{i,j}) x_{i,j} , \forall i \in F, \forall j \in N \cup F \tag{11} \)

\( e_{i,j} - e_{k,i} \leq (-M - b_{i,j}) x_{i,j} + M, \forall i \in N_0, \forall j \in N \cup F \tag{12} \)

\( b_{min} \leq e_{i,j} \leq b_{max}, \forall i, j \in N \cup F \tag{13} \)

The objective function is defined in Equation (4) and presents two terms which must be minimized: the travel time cost and the recharging time cost, which is considered only when a charging station is visited.

Equation (5) and (6) states that the initial node 0 must be the starting and ending node of the tour. Equation (7) guarantees that each Target is visited exactly once. Constraint (8) preserves the flow conservation, each visited vertex must be an arriving and starting vertex. Equation (9) defines the subtour elimination constraint, as introduced in [16]. With subtour we mean a closed path, generated by a cluster of targets and CS nodes, which is isolated from the principal tour starting from the initial node. \( S_v \) is the set of vertices contained in an existing subtour \( v \) in the solution.

Constraint (9) is repeated in the constraint matrix \( A \) as many times as the number of subtours \( S_v \) are present in solution. Equation (9) imposes that the number of the all possible arcs \((i,j)\) between the vertices that compose the subtour \( S_v \) must be lower equal than \(|S_v| - 1\), where \(|S_v|\) is the dimension of set \( S_v \). Since it is impossible to know in advance if the solution could present inner subtours, the dimension of the set is initialized to a sufficiently big number \(|S_1| = |N| \times |F|\), greater than all the possible arcs combination in the solution. The iterative Optimization process is described in Algorithm(1), visible below. In particular, the function \texttt{detectSubtours} takes as input the optimal path founded by the solver. Knowing the list of segments between the vertices (Target and CS visited) that compose the solution, a subtour is detected exploiting the properties of the Adjacency matrix, which is built from the vertices visited in solution. This matrix, by definition, permits to known if a graph is fully connected and the segments involved in each path length. From the properties of the Adjacency matrix it is possible to find if different subtours are present in the solution, and then proceed with a new call to the solver function only after equation (9) is updated inside the function \texttt{OptimizationProblemDefinition}.

\begin{algorithm}
\caption{Optimization Algorithm description}
\begin{algorithmic}[1]
\State Initialization:
\State \( \Sigma = \{S_1 = N \cup F\} \);
\State numtours = 1;
\State do
\State 5 OptimizationProblemDefinition(\( \Sigma \), numtours);
\State 6 \( \text{path} = \text{callToOptimizationSolver} \);
\State 7 \( [\Sigma, \text{numtours}] = \text{detectSubtours(path)} \);
\State 8 while numtours > 1;
\end{algorithmic}
\end{algorithm}
Finally equation from (10) to (13) model the energy dynamic of the UAV, where M is a large number and $b_{i,j}$ is the estimated battery consumption on each arc. Equation (14) and (15) define the MILP problem with the integer and continuous variables. The utilization of only two decision variables is adopted to increase the performance of the solver and reducing the computational time.

IV. EXPERIMENTAL RESULTS

Two types of experiment were conducted. The first one, described in section IV.A, is based on iterative simulations of various scenarios with different number of vertices. In each test the number of the target nodes is the same of the number of CS nodes and are randomly placed in a Square Area of 144 km$^2$. Each simulation with the same number of vertices is repeated ten times and the data obtained from the tests are analyzed in order to understand how the number of vertices, assuming an equal number of targets and CS nodes, can affect the optimization time $t_{opt}$. Furthermore, computational and energetical performances have been analysed in a different scenario, where the number of targets was different from the number of CS.

In section IV.B a field patrolling scenario it is proposed in a countryside region located in Northern Italy. Some target fields, marked with a red square in Figures 6 and 7, need a detailed inspection which requires a certain time $t_{ins}$. The CS, visible as blue circles in Figures 6 and 7, are located near the houses were the residents volunteers have granted their availability to take care of the drone if it lands on their property to be recharged. Different tests were conducted to find the maximum inspection time $t_{ins}$ in each target given an increasing distribution of CS on the territory. The experiments were conducted using the MATLAB® Optimization toolbox, using the MILP solver based on the Branch and Bound Algorithm and CPLEX method. The Program is run on an Intel 15 processor with 8 Gb of memory.

A. COMPUTATIONAL PERFORMANCE ANALYSIS OF THE USRCP

In this section the first experiment is proposed and analysed.

In the first experiment different $k = 7$ worlds have been generated, with a total number of vertices equal to $|N_k \cup F_k| = [4, 8, 12, 16, 20, 24, 28]$ (plus the initial node) and with an equal number of targets and CS nodes in each scenario. For each scenario $k$ a series of $j = 10$ different tests are performed, e.g.: the simulation with a scenario composed of 4 vertices (2 Targets, 2 CS nodes) is repeated 10 times. At every iteration the location of the vertices is defined randomly. In particular, in the graph shown in Figure 2, we evaluate the mean and the standard deviation of the optimization time $t_{opt}$, respectively indicated as $t_{opt}$ and $\sigma_{t_{opt}}$, requested by the solver to find a solution with respect to the vertices number in each of the generated scenarios $k$.

As visible in Figure 2, the mean of the optimization time evaluated on the set of generated worlds tends to increase with the number of vertices in the simulation. A great contribution to the exponential increment of the optimization time is surely also given by the presence of subtours, since if they are detected in the solution, it is required to repeat a more constrained optimization until a new solution with only one tour is found. This information is provided by the variance, which is greater in the simulations set with 24 and 28 vertices. The higher number of nodes increase the probability to cluster generation which can lead to find an optimal solution with multiple separated paths. Indeed, in the last two scenarios $k$, it was necessary to repeat two tests, because it was detected a total number of five subtours in the ten tests performed. For example, test $j = 3$ and test $j = 6$ in the last scenario have been presented respectively $|\Sigma_3| = 2$ and $|\Sigma_6| = 3$, where $|\Sigma_j|$ is the dimension of the set $\Sigma$.

The results of a different experiment, keeping the number of vertices constant (i.e., 24), but having different ratio $r_k = |N_k| / |F_k|$ between the number of target and CS nodes, are shown in Figures 4 and 5.

A total number of $k = 9$ different tests were analyzed, each one with $|N_k| + |F_k| = 24$ vertices and respectively with $r_k = [4/20, 6/18, 8/16, 10/14, 12/12, 14/10, 16/8, 18/6, 20/4]$. This experiment is conducted in order to analyze the influence of the distribution ratio $r_k$ on the optimization time $t_{opt}$ required to find a solution and the battery consumption. Figure 4 shows the optimization time $t_{opt}$ required to solve each scenario with respect to the ratio $r_k$ and the number of subtours explored by the solver.

As expected, the $r_k$ value has an important impact on the computational time required by the solver to find the optimal path. As the $r_k$ value increases, the optimization time $t_{opt}$ increases exponentially. This is due because the system is more constrained in energetic terms, and during the Branch and Bound process a less number of solutions can be discarded. Thus, the algorithm approaches the worst case, which consists in analyzing all possible permutations.

Figure 5 represents the total distance traveled by the UAV in each scenario $k$ with respect to the mean battery consumption, defined as $\bar{e}_k$, to complete the path. As expected, the distance traveled increases with the number of targets explored. On the other hand, the mean of the battery level, which is constrained between $b_{max} = 1$ and $b_{min} = 0.3$, seems not to be particularly conditioned by the varying length of the path in each test. Indeed, in the 88.88% of the test analyzed, the mean battery value $\bar{e}_k$ is contained inside the range 0.7 < $\bar{e}_k$ < 0.5.

Colored regions in Figure 5 indicate the number of CS visited to complete the path. A very interesting result is visible in test $k = 9$, where the distance required to complete the path is greater than 70 Km but it is necessary to visit only five stations to recharge the batteries. Knowing that $r_g = 5$, we obtain that the size of $|F_g| = 4$. This means that a recharging station is visited more than one time.

Finally from these series of tests, it is evident that the global number of CS visited in the solutions is only the 19.8% of those available in the experiments conducted with different...
In conclusion, the experiments presented in this section give some detailed information about the influence of three main factors on system performance. In particular, we have seen, until this point, that the computational time $t_{\text{opt}}$ is heavily influenced by:

- The number of vertices.
- The distribution of the vertex, causing the possible generation of subtours to solve.
- The relation between the number of Target and CS nodes.

B. Simulation of a Real Patrolling and Inspection Scenario

The idea here is that $T$ is the set of targets to be monitored, which are supposed to be given in advance because they correspond to significant locations to be repeatedly patrolled during the year (i.e., houses, emergency medical services, bridges, level crossings, etc.). $F$ is the set of all feasible locations to set up a charging station, located near houses. However, this time, authorities will not consider all possible locations for setting up a charging station: instead, they will select, if possible, a smaller set $A \subseteq F$ to involve a smaller number of residents in the process (e.g., because involving residents has a cost in terms of insurance and other bureaucratic reasons). Considering the set of selected charging station $A$, we will evaluate the optimal path by increasing the inspection time spent in each Target. Specifically, the inspection time is increased at each simulation, until no solutions are found given the current distribution of the available charging stations spots. In this case, a new series of iteration is done, with the availability of additional charging stations.

In particular the initial inspection time for each Target is fixed at $t_{\text{is}} = 5\min$. At every iteration is increased of $\Delta t_{\text{is}} = 1\min$. The initial dimension of the set is $A = 6 \leq F$. If no solutions are possible given the current value of $t_{\text{is}}$, the set $A$ is expanded to include two more charging stations until no more available charging spots can be selected ($|A| = |F|$). Figure 6 and 7 show the optimal path between targets (red squares), computed with $|A| = 6$ (blue circles) and $t_{\text{is}} = 5\min$, and the path obtained with the maximum number of CS spot available $|F| = |A| = 14$ and a maximum inspection time of $t_{\text{is}} = 1080\sec$.

A more detailed explanation of the simulations is visible...
TABLE I: Results of the patrolling and inspection scenario

| N° test i | |A| | N° CS visited | t_{is} [s] | t_{opt} [s] | ¯ei | Zi [sec] | Optimal Path |
|----------|-------|----------------|-------------|-------------|--------|-------------|---------|-----------|---------------|
| 1        | 6     | 2              | 300 s       | 0.9949 s    | 0.6485 | 1842.6497 s | 1-5-6-4-3-2-1 |
| 2        | 6     | 2              | 360 s       | 1.5313 s    | 0.5876 | 2082.6497 s | 1-5-6-4-3-2-1 |
| 3        | 6     | 2              | 420 s       | 1.1918 s    | 0.5343 | 2573.5443 s | 1-5-6-3-8-4-6-2-1 |
| 4        | 6     | 3              | 480 s       | 4.7405 s    | 0.6263 | 3506.9219 s | 1-5-6-3-8-4-6-2-1 |
| 5        | 6     | 3              | 540 s       | 2.2349 s    | 0.5172 | 3746.9219 s | 1-5-6-3-8-4-6-2-1 |
| 6        | 6     | 3              | 600 s       | 2.5454 s    | 0.3770 | 4706.9219 s | 1-5-6-3-8-4-6-2-1 |
| 7        | 8     | 3              | 660 s       | 1.4687 s    | 0.6956 | 1988.2101 s | 1-7-5-4-8-3-2-1 |
| 8        | 8     | 4              | 720 s       | 3.9803 s    | 0.5925 | 3773.5862 s | 1-2-8-4-12-3-6-5-1 |
| 9        | 8     | 4              | 780 s       | 8.2193 s    | 0.3693 | 5709.8540 s | 1-5-10-3-8-4-6-2-1 |
| 10       | 10    | 2              | 840 s       | 2.2202 s    | 0.5050 | 2706.8776 s | 1-5-4-8-3-2-1 |
| 11       | 10    | 4              | 900 s       | 14.8524 s   | 0.4232 | 5198.0493 s | 1-5-6-3-8-4-14-2-1 |
| 12       | 12    | 4              | 960 s       | 3.5168 s    | 0.6094 | 3175.6543 s | 1-5-8-4-16-3-2-15-1 |
| 13       | 12    | 5              | 1020 s      | 6.7561 s    | 0.4013 | 6119.5881 s | 1-2-17-8-4-16-3-6-5-1 |
| 14       | 14    | 5              | 1080 s      | 5.9373 s    | 0.3756 | 6229.5881 s | 1-2-17-8-4-16-3-6-5-1 |

Table 1. Each row presents the data coming from the test $i$, by reporting the number of CS available (set $A$), the number of CS visited to find a solution, the inspection time $t_{is}$ considered, the computational time $t_{opt}$ required to find a solution, the average battery $\bar{e}_i$, the optimal cost $Z$ expressed in time to complete each path and finally the sequence of vertices visited in the optimal path.

From the $\bar{e}_i$ column it can be seen how the average battery decreases while the inspection time $t_{is}$ linearly increases despite a constant number of available CS (until a solution cannot be found, and the number of CS is increased as well). Obviously, the battery consumption value is influenced by the shape of the path and consequently by the number of CS visited. This is visible looking to the data from test $i = 13$ and test $i = 6$. Despite the value of $Z_6 < Z_{13}$ (i.e., the flight time requested to complete the inspection is lower in test 6 than in test 13), the relation that holds between the average battery consumed is $\bar{e}_6 < \bar{e}_{13}$. This is due by the distribution of the CS on the operative area. In fact, the analysis of the order of the visiting vertices in each path as reported in the last column with the disposition of all vertices in Figure 7, confirms that the distribution of the CS on the area permits to increase the flight time reducing the average battery consumed.

V. CONCLUSION AND FUTURE WORK

This paper proposes a novel approach for the optimal routing of UAVs based on the new concepts of Social Charging Stations and Social Drone Sharing. Indeed, involving citizens in the monitoring process may be vital for increasing UAVs patrolling capabilities. The approach has been preliminarily validated with simulation experiments, analysing computational time and general performances, and eventually showing its feasibility.

Future extensions of this work concern the possibility of using more drones that share the Targets and their inspection time.

In any case, the possibilities to expand this work, to adapt it to a real patrolling scenario, are enormous, and all very interesting to explore, ultimately leading to a wider deployment of UAVs in critical contexts.

REFERENCES

[1] Patrik Schmuck and Margarita Chli, “Multi-UAV Collaborative Monocular SLAM,” 2017 IEEE International Conference on Robotics and Automation (ICRA) Singapore, May 29 - June 3, 2017.

[2] Nadeem M Shajahan, Arjun Sasikumar, Thomas Kuruvila and Dhivin Davis, “Automated Inspection of Monopole Tower using Drones and Computer Vision” 2019 International Conference on Intelligent Autonomous Systems (ICoAS).
[3] https://www.neimagazine.com/features/featurefukushima-looks-to-drones-6728269/
[4] https://www.thelocal.it/20200410/hovering-police-drones-take-italians-temperature-and-issue-fines.
[5] Y. Chen, D. Baek, A. Bocca, A. Macii, Enrico Macii, M. Poncino, “A Case for a Battery-Aware Model of Drone Energy Consumption,” 2018 IEEE International Telecommunications Energy Conference (IN-TELEC), vol. A247, pp. 529–551, October 2018.
[6] https://www.dji.com/it/mavic-2/info.
[7] David L. Applegate, Robert E. Bixby, Vašek Chvatá and William J. Cook, “The Traveling Salesman Problem: A Computational Study,” Princeton, NJ, USA:Princeton University Press Eds. New York: Academic, 2011.
[8] J. C. Escobar, H. A. González and P. C. García”Quadrotor Quaternion Control”, 2015 International Conference on Unmanned Aircraft Systems, June 2015.
[9] Phuong D. H. Nguyen, Carmine T. Recchiuto, Antonio Sgorbissa “Real-Time Path Generation and Obstacle Avoidance for Multirotors: A Novel Approach”, Journal of Intelligent Robotic Systems, 24 January 2017.
[10] Khoufi, I.; Laouiti, A.; Adjih, C. A Survey of Recent Extended Variants of the Traveling Salesman and Vehicle Routing Problems for Unmanned Aerial Vehicles. Drones 2019, 3, 66.
[11] Quang Minh Ha,Yves Deville, Quang Dung Pham, Minh Hoàng Hà "On the Min-cost Traveling Salesman Problem with Drone” Preprint submitted to Transportation Research Part C: Emerging Technologies November 24, 2017
[12] Murray, C.C.; Chu, A.G. The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery. Transp. Res. Part Emerg. Technol. 2015, 54, 86–109.
[13] Sungwoo Kim, Ilkyeong Moon, "Traveling Salesman Problem With a Drone Station,” October 2018, IEEE Transactions on Systems, Man, and Cybernetics: Systems PP(99):1-11.
[14] Sai Shao, Guan Wei, Bin Ran, Jun Bi, Zhengbing He , “Electric Vehicle Routing Problem with Charging Time and Variable Travel Time,” January 2017, Mathematical Problems in Engineering 2017(2):1-13
[15] Khuller S., Malekian A., Mestre J. (2007) To Fill or Not to Fill: The Gas Station Problem. In: Arge L., Hoffmann M., Welzl E. (eds) Algorithms – ESA 2007. ESA 2007. Lecture Notes in Computer Science, vol 4698. Springer, Berlin, Heidelberg
[16] Pferschy, U., Staněk, R. Generating subtour elimination constraints for the TSP from pure integer solutions. Cent Eur J Oper Res 25, 231–260 (2017). https://doi.org/10.1007/s10100-016-0437-8
[17] J. Gagoa, C. Doutheb, R.E. Coopmanc, P.P. Gallegoa, M. Ribas-Carbob, J. Escalonab, H. Medranob. (2015) “UAVs challenge to assess water stress for sustainable agriculture”. Agricultural water management, 153, 9-19.
[18] Chien-Ming Tseng, Chi-Kin Chau, Khaled Elblessioni, and Majid Khojii. 2017. Autonomous Recharging and Flight Mission Planning for Battery-operated Autonomous Drones. 1,1,Article1(January2017),25pages.
[19] Recchiuto, C.T., Sgorbissa, A., Zaccaria, R. Visual feedback with multiple cameras in a UAVs Human-Swarm Interface (2016) Robotics and Autonomous Systems, 80, pp. 43-54.
[20] Iacono, M., Sgorbissa, A. Path following and obstacle avoidance for an autonomous UAV using a depth camera (2018) Robotics and Autonomous Systems, 106, pp. 38-46.
[21] Recchiuto, C.T., Sgorbissa, A. Post-disaster assessment with unmanned aerial vehicles: A survey on practical implementations and research approaches (2018) Journal of Field Robotics, 35 (4), pp. 459-490.
[22] https://mashable.com/article/carplay-carkey-ev-apple-maps-wwdc-2020/?europe=true