Forest Fire Monitoring Through a Network of Aerial Drones and Sensors

D. Simões, A. Rodrigues, A. B. Reis, S. Sargento
Instituto de Telecomunicações, Universidade de Aveiro, Aveiro, Portugal
Email: {danielasimoes, andrenascimento, abreis, susana}@ua.pt

***Abstract*—With the technological advances in recent years, elements such as aerial drones and its sensors are becoming increasingly accessible, smaller, transportable and also more stable in flight. Aerial drone fleets are ideally suited to monitor an active forest fire and to support firefighting teams on duty.**

In this paper we propose a system to support fire monitoring, built using a Flying Ad-hoc Network (FANET) and Unmanned Aerial Vehicles (UAVs), commonly called aerial drones, focused on collecting data from firefighters on duty, in wildfires' context. This system provides the sensing of a fire perimeter, and monitors and follows the firefighters while they are on duty. This system is validated both through simulation and through real-life experiments performed with a FANET of three UAVs. The results obtained in these experiments show that the proposed algorithms have good performance, and that the corresponding system is able to support the firefighting mission in an autonomous approach.

***Index Terms*—Flying Ad-hoc Networks, Unmanned Aerial Vehicles, Firefighting**

I. Introduction

Early detection and fast response of the fire departments is crucial to ensure that forest fires are extinguished quickly and cause minimal economic damage and loss of life. Land-based systems, however, are limited in their mobility and land coverage; satellite systems have issues with many simultaneous hotspots; and manned air vehicles are usually expensive, and also carry a risk to the pilots' lives.

Unmanned Aerial Vehicle (UAV), also known as drones, have been considered as an option to either substitute or cooperate with pre-existing methods. UAVs are usually cheaper, faster, effective and can be used to reach hazardous areas, for example to perform radiological mapping and monitoring of scenarios with nuclear threats [1]. This paper proposes a system to support firefighting through the use of a network of aerial drones. In this system it is proposed an algorithm that uses a drone and its sensors to delineate a fire area – a forest fire. Moreover, drones are autonomously forming a network to help on the firefighters rescue, by monitoring them and providing the possible data for a successful and effective rescue. Finally, it is proposed an ad-hoc network that flies over firefighters in action, making decisions based on the data they gather from firefighters, and strategically position themselves to obtain the information and provide it to a ground station. The ground station contains a dashboard where it is possible to manage the drones, view the collected data in list or on the map, in case of locations, and help on the firefighters mission.

The proposed approach was able to delimit a fire area with 404 squared meters using only temperature and GPS sensors in 4 minutes, and to monitor firefighters and verify multi-hop behavior in the network, as well as adjusting UAV positions to gather the most data from firefighters. When in rescue, the UAV was able to arm and reach a position near enough to collect data of the firefighter, who was 19 meters away, in less than 1 minute.

The remainder of this paper is organized as follows. Section II presents related work on the use of drones in emergency patrols and related scenarios. Section III describes the architecture used to build the system, and Section IV presents the implementation of the proposed algorithms. Section V shows the real-life experiment results with the integration of a routing protocol. Finally, Section VI provides concluding remarks.

II. Related Work

Several works have proposed the use of UAVs to improve operations in various areas. The authors of [2] propose the use of UAVs to help in searching regions where the rescue teams may have better chances of finding victims. The work in [3] aims to allow UAVs to perform intelligent tactical movements in a disaster scenario, combining Jaccard distance and artificial intelligence. The main purpose is to maximize the number of victims that are served by the UAVs and avoid network disconnection.

In [4] the goal is to develop a system of surveillance, where the drone must take-off and land automatically. Each drone communicates with the control server via the internet, and the server provides a patrol service in a predefined area. The work in [5] describes a system for monitoring a fence about an ad-hoc perimeter: using a movement detection mechanism, a signal is transmitted in response to at least one of the autonomous drones detecting a violation or intrusion of the ad-hoc perimeter. The work in [6] develops sensor packages for small robots that can determine the path of the leader and to develop path following algorithms. This work uses a carrot-following approach to path following: the carrot position is updated every cycle by the path following algorithm, and is located by finding the intersection of a circle and the extended path segment.

There are, notably, few solutions designed specifically to support firefighters during wildfires, and even fewer that require no supervision. The use of UAVs should be mostly autonomous, requiring only one firefighter to activate brief tasks (e.g., launching, recharging) and monitor the data to make decisions in a more informed way. The aim of this work is to develop such a solution, where the drones manage their own network, find the ideal positions to collect and deliver data from as many firefighters as they can, avoid the fire, and do these tasks autonomously.

III. Architecture

The architecture considers UAVs connected directly, or though other UAVs, to a Ground Station (GS).

A. Ground Station

The GS must be physically close to the firefighters. We assume that the GS will stay in the fire truck. To interact with the GS, a simple dashboard was built. The main features provided by the Dashboard are: map visualization with drone, firefighters and ground station; visualize drone and firefighters alert when in danger; drone path visualization; init and cancel missions; visualize sensor data.

The GS uses a WebSocket to receive information from and provide information to the dashboard. WebSocket was chosen because of its low latency and reliability. There is also a Drone Logs module which stores the logs of the GS. This module stores every message, from location messages to drone status. This module communicates with the modules Store Drone Path and Drone Register and feeds information for these two modules. The Store Drone Path module stores the coordinates sent by the drone which define the fire perimeter. The Drone Register stores the drones' IDs, their type (monitor drone, follow drone or perimeter drone) and initial positioning; this module is used to store this information once and queried later to write information in the dashboard. The Drone Proxy Commands module redirects the commands sent by the dashboard to the drones. Finally, Robot Operating System (ROS) module is a simple ROS2 instance which builds the bridge between the GS and the drones.

B. UAVs

The UAV has its own sensors, in the current case, temperature (for safety reasons) and GPS. The values of those sensors are sent via ROS2 to the GS. The ROS2 is also used to receive commands such as "cancel mission", or "init mission perimeter". These commands are parsed and then the drone logic takes action through DroneKit [8], a Python API that communicates with the flight controller over MAVLink [7], providing mission management and control over the UAV. The MAVProxy forwards the MAVLink messages to the flight controller and in result, it is possible for the UAV to perform its assigned missions as a set of consecutive commands.

C. Firefighter

We assume that firefighters are equipped with temperature sensors and a GPS receiver, at a minimum. This way it will be possible to track the firefighters' positioning in the field. This data is sent via ROS2 to the UAV ROS2 instance, and then forwarded to the GS.

IV. Implementation

A. Monitoring Mission

The fire conditions change very quickly, forcing the firefighters to adapt their positions on the field. However, it is also common for the firefighters to be distributed near the firehose, meaning that they are mostly distributed in groups. It is very likely that firefighters are near each other, thus the FANET should reflect that. Therefore, it was decided to build only one FANET, encompassing the largest number possible of firefighters. This network does not ensure that all of the firefighters are covered by the network, but ensures that the network will expand while firefighters lacking coverage exist.

The main goal of this FANET is to adjust the drones' positions automatically according to the firefighters' positions on the field. This way it will be possible to collect the data from the largest number of firefighters possible, without losing the connection between drones. This lays on the fact that, collecting information in a volatile environment is only useful when the delay is reasonably low (30-60s at most), to make sure that the data from the firefighter will not be outdated. While the drone has a mechanism to store information when disconnected, it is only useful in some scenarios: for example, in the mission to set the perimeter it is very likely that the drone will not have connectivity all the way. In missions of surveillance, however, this is not helpful because the collected data is time-critical and cannot be delayed. It would be counterproductive to use a system which would endanger the whole operation – thus, the UAVs try to never disconnect from the network while in surveillance and monitoring missions.

The implementation of the algorithm was inspired on [4] with notable improvements such as the method to calculate the target Jaccard distance, the algorithm to move the drones, and the context itself. While the purpose of their project is to help victims in an emergency scenario, the purpose of this work is to bring information in a forest fire scenario to the Ground Station, in which firefighters do not represent a random distribution (unlike victims of a tragedy). Yet, there are similarities such as the concept of neighbors and Jaccard Distance.

1) Neighbors: The concept of neighbors describes the UAVs that have influence in another UAV. The UAVs will move according to the firefighters positions and movement through a Jaccard Distance: to move the UAV farther away, meaning that the FANET must expand because there are firefighters not covered, or to move the UAV closer, because the network is in risk of disconnection. The number of neighbors to select is 2, due to vector angle cancellation.

The UAVs which will be assigned as UAV neighbors are the ones with the minimum Jaccard Distance. This means that every UAV will calculate the Jaccard Distance between itself and all the UAVs it is possible to have a connection with and select the 2 with the smallest Jaccard Distance. The assigned neighbors are not static, and according to the FANET arrangement, new neighbors will be assigned to one UAV.

2) Jaccard Distance: The Jaccard Distance measures the dissimilarity between two samples, which is complementary to the Jaccard Coefficient (which measures similarity between two groups). It is given by:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}. \quad (1)$$

The Jaccard Distance is given by:

$$d_J(A,B) = 1 - J(A,B) = \frac{|A \cup B| - |A \cap B|}{|A \cup B|}, \quad (2)$$

where $J(A,B) \in [0,1]$. The Jaccard Distance will measure how many firefighters are being served by the same UAV. This way, it will be possible to expand the network when there are many firefighters being served by the same UAV, and vice-versa. It is assumed that firefighters are close to the UAVs when the mission begins, so UAVs can start receiving their data from the start in order to perform the monitoring mission.

3) Target Jaccard Distance: The target distance is the *ideal* distance to maintain. This is the metric that each UAV will use to take the decision to apply a repulsion force or an attraction force in a neighbor. Initially, the target distance is set to 0.5, and according to the FANET and firefighters distribution, this value will also adapt in order to spread the network or to retract it. If not all firefighters are covered, it means the network must spread, so the value is increased in 0.1; in the opposite scenario, if there are no more firefighters to be covered, it means the network can retract slightly – so if the mean of the Distance of Jaccard of both neighbors is inferior to the initial value (0.5), the value is decreased in 0.1; else, the target value does not change. It is worth mentioning that the target distance is truncated between 0.3 and 0.7. This way, the UAVs will not spread enough to lose connectivity between themselves, and will not aggregate enough to crash into each other.

B. Fire Perimeter with Semi-Autonomous Flight Mission

The developed perimeter algorithm (algorithm 1) does not contemplate complete autonomous flight, as it is out of the scope of this work. An anti-collision system was developed to avoid collision between drones, but not other obstacles, so the assumption is that the UAVs fly at such a height that they do not collide with trees and other obstacles.

This algorithm is based on the metric of **temperature**, takes action based on geometric angles, and works in a clockwise direction. The UAV also stops automatically and returns to home when reaching a point near the first location. The following variables are used:

- **last_go_angle, go_angle**: stores the last adopted angle and the angle to be adopted, respectively.
- **last_temperature, temperature**: stores the last temperature and current temperature measured by the UAV sensors' when the UAV decides its new position.
- **delta_temperature**: the difference between the last temperature and the current temperature.
- **go_distance**: the size of the step to be taken by the UAV, set to 30 meters.
- **safe_max_temperature**: stores the maximum temperature considered safe to the UAV flight, with the value of 55 degrees celsius.

```
safe_max_temperature ← 55
go_distance ← 30
last_think_time ← 0
while true do
    elapsed_time = ELAPSEDTIME(LAST_THINK_TIME)
    if elapsed_time ≥ 1 then
        if next_way_point = last_next_way_point then
            last_next_way_point ← next_way_point + 1
            delta ← temperature − last_temperature
            factor ← abs(safe_max_temperature − temperature)
            df ← delta/factor
            go_angle ← last_go_angle + df ∗ last_go_angle
            if go_angle ≤ 120 then
                go_angle ← go_angle+ = 360
            else if go_angle ≥ 520 then
                go_angle ← go_angle− = 360
            end if
            delta_distance ← base_delta_distance − delta
            MOVEVEHICLE(GO_ANGLE,GO_DISTANCE)
            last_go_angle ← go_angle
            last_temperature ← temperature
            last_think_time ← TIME.NOW()
            if ARRIVEDINITIALPOSITION() then
                break
            end if
        end if
    end if
end while
```

Alg. 1: Monitoring firefighters.

- **last_think_time**: stores the last time when the logic was executed.

C. Rescue Mission

The rescue mission, Fig.1, assumes that the FANET is already operating, receiving data from the firefighters and exchanging this data between UAVs. When the data from a firefighter presents values above a predefined threshold, the rescue mission is activated, sending one UAV to the firefighter's location and keeping tracking of the firefighter until the mission is cancelled or a defense mechanism is activated. When the battery runs low, a message is broadcast to every UAV, and a UAV with no mission assigned takes the tracking role.

D. UAV Self-Defense Mechanisms

1) Fire Avoidance: The Fire Avoidance mechanism stops the UAV from approaching areas where the temperature is unsafe for UAV. This means that UAV safety has priority over any assigned mission: while on a mission, the UAV keeps measuring temperature, and if it exceeds a set threshold, it will calculate the inverse path in a way that allows it to continue to sense the firefighters safely.

2) Drone Replacement: When the battery of one UAV reaches 40% or less (user-configured), a message is sent to the Ground Station requesting another available UAV to take its place. This exchange only happens when a UAV is available with no mission assigned. If no new UAV is available, the UAV simply returns to home to recharge.

3) Anti-Collision System: The Anti-Collision System (Fig.2) calculates if there is a UAV in the path between one UAV's current and next positions. The algorithm adds an extra margin to the path of 0.5 meters to each side. If a UAV is near but not crossing the line, there is no harm, because the safety area is therefore a rectangle with 1 meter of width and a length between its current and next position.

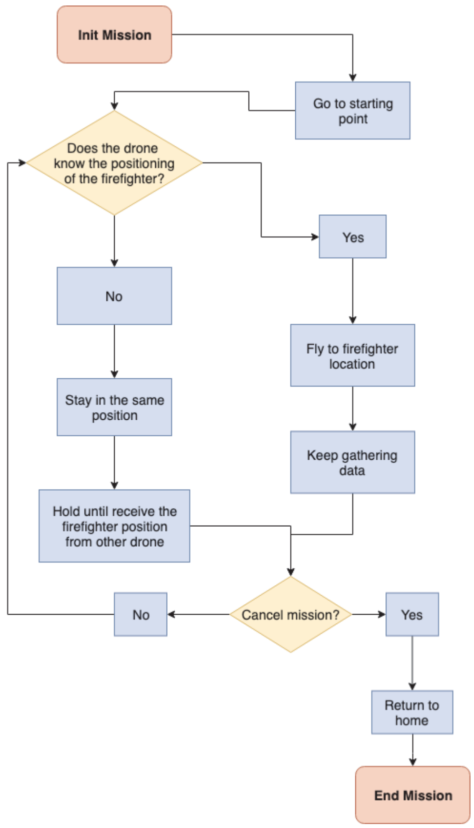

Fig. 1. Rescue mission process.

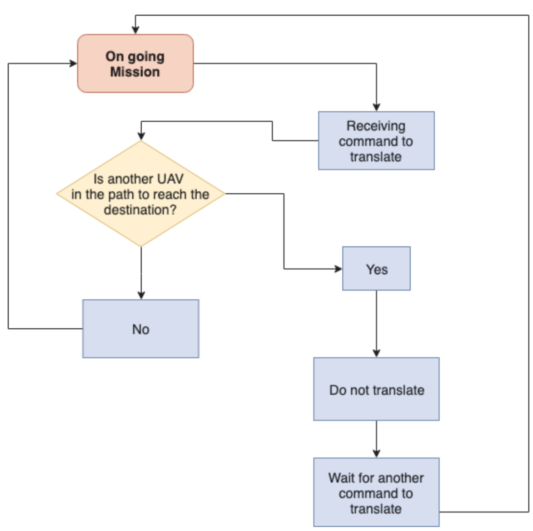

Fig. 2. Anti-collision process.

If a UAV is in the path that another UAV has calculated, it does not translate and waits for a new command. If the new command is the same, and the other UAV is no longer in the path, the translation proceeds, however, the FANET may change and the command will eventually be different (with another destination). While the UAV waits for another command, the network will not stall, because the firefighters keep moving and other UAVs also keep moving, causing the network to adapt and not obstruct the path for an excessive time.

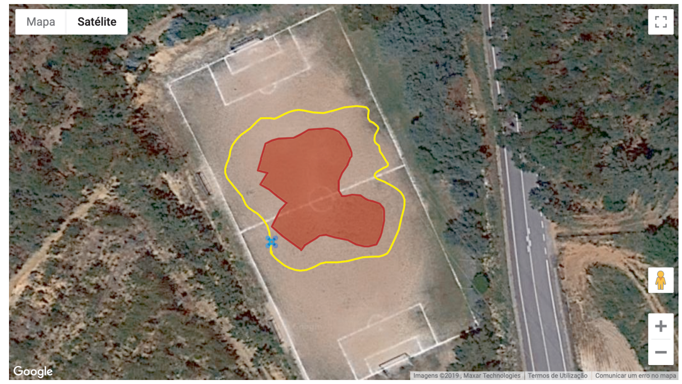

Fig. 3. Perimeter algorithm result.

V. Functional and Performance Results

This section describes experiments taken in a football field, where a fire fence was simulated with a temperature gradient to test the behavior of the UAVs.

A. *Perimeter Mission*

In this experiment, shown in Fig. 3, the expected behavior is for the UAV to outline the simulated fire. In the first test, the fire assumes a round shape with $285m^2$ of area and 65m of perimeter. The UAV behaved as expected, outlining the fire, and ending its mission where it detects to be in an area that it already outlined. As a result, the UAV flew approximately 129 meters in roughly 4.5 minutes with a velocity of 0.5m/s.

In a shape with more randomness, with $404m^2$ of area and 99m of perimeter (not shown), the UAV starting point is already very near to the fire fence with the goal of saving battery. It is possible to verify that the drone behaved as expected, and finished its mission successfully. The UAV flew approximately 121 meters in 4 minutes with a velocity of 0.5m/s. One can conclude that the algorithm has better performance in larger fire perimeters. This fact is justified by the angles of flight – in a larger fire perimeter, the UAV does not need to take sharp turns.

B. *Rescue Mission*

The rescue mission is aimed to be activated when something is wrong with a firefighter – e.g, a sensors' value is above a threshold. Heart rate and other health-related sensors are one example, but for the purpose of this test, only temperature metrics were considered. The drone flew over the specific firefighter until the mission was terminated, showing that the UAV behaved as expected. It flew approximately 19 meters to the firefighter, which kept moving. The UAV reached a close enough position to start receiving data from the firefighter in roughly 1 minute since the mission was assigned.

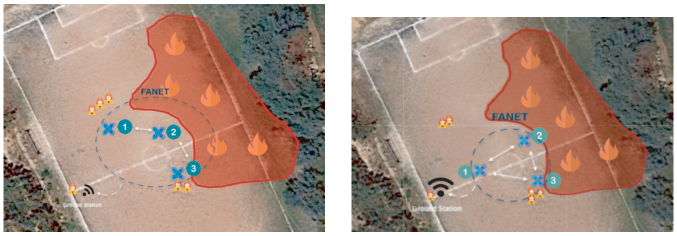

Fig. 4. Multi-hop scenario when the formed triangle is wider (left), where UAVs have multi-hop techniques to communicate, and non-multi-hop scenario (right) where UAVs are directly connected.

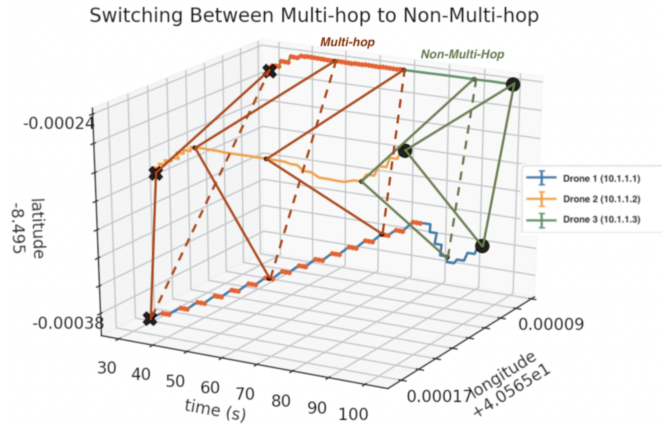

Fig. 5. Drone trajectories between multi-hop stage and non-multi-hop stage, marking active routes.

C. Routing Protocol Integration

For UAVs to receive data from the firefighters and pass that data to the Ground Station, they may need to send data over another hop or several hops (multi-hop communication). This work was integrated with an existing routing platform with a version of Greedy Perimeter Stateless Routing (GPSR) improved with position prediction. One of the goals of the previously-mentioned missions is to provide the best wireless coverage area by forming the best mesh placement, limited to the formation made by the number of UAVs available. Through this integration the drone shares its attributes throughout the network, informing foreign nodes so they can correctly establish a route towards this node. The new routing version learns the network topology by itself and predicts each neighbour's position to find a next-hop that is the optimal path between a pair of nodes.

In an initial stage, teams are split. UAVs distance from each other to provide wireless coverage to both teams, and, once the FANET maximizes the coverage area without losing connection, the triangle formed by the UAVs will extend its vertexes to cover both teams and still keep a connection with the Ground Station, and then, on this condition, the routing platform performs multi-hop inside the FANET, as depicted in Fig. 4, on the left image.

Then, on a later stage, firefighting teams get closer to each other, then, given three UAVs, the topology formation reveals a small triangle where every UAV is a neighbor, as depicted in Fig. 4 on the right image.

In between these two described scenarios, the behavior of UAVs, their position, and active routes are registered. The trajectory of each UAV is traced on Fig. 5, where the black 'X' marks the beginning of the path (multi-hop stage), the black 'O' symbolizes the end (non-multi-hop stage), the red dot marks an existing active route. The red triangle symbolizes the distance between UAVs from which it can be experienced multi-hop communications. The green triangle symbolizes the distance between UAVs on its last iteration, then they appear to be closer and experience non-multi-hop communications. Over time, this experiment shows that initially the triangle made by UAVs is larger and becomes smaller, meaning that drones come closer to each other; thus it is possible to divide into different time zones whether UAVs adopted multi-hop.

The previous scenario describes the expected results and explain how UAVs interact with each other, and then confirms the expectations with the logging results presented on each stage. On the experiment initial phase, UAVs on the multi-hop stage are distanced from each other, resorting to multi-hop to communications; in this case, drone 2 acts as a bridge for communications between drone 1 and drone 3, as previously described. Over time UAVs become closer to each other, so multi-hop is not needed anymore.

Each UAV calculates the Jaccard Distance between itself and its neighbors. Considering the already explained algorithm and the covered firefighters, the Jaccard Distance result can be depicted in Table I. Considering that the maximum target value is 0.65, it means the network must start to retract. Considering that all the firefighters are covered and the values 0,7 and 0,75 are superior to 0,65, the drone 2 and the drone 3 will advise the drone 1 to come closer; then the drone 1 calculates the angle and starts moving closer until reaching the position depicted in the second stage of this experiment.

VI. Conclusion

In this paper we proposed an autonomous network of UAVs to support firefighters in a wildfire scenario, which collect data from a variety of sensors and provide it to the Ground Station. A FANET adjusts the UAV's positions in order to provide coverage and sink the gathered information to the Ground Station, where a dashboard makes it possible to see UAVs and firefighters, manage missions and visualize the received data.

Through an algorithm based on Jaccard distance, we proposed a monitoring approach to locate and move UAVs in an autonomous way to follow firefighters. These developments

TABLE I
Jaccard Distance Calculation

	JD Drone 2	JD Drone 3	Covered Firefighters
Drone 1	0,7	0,75	60,61,62,63
Drone 2	–	–	62,63,71,72,73
Drone 3	–	–	62,63,70,71,72,73

have been tested in the field and integrated with a routing protocol developed with the goal of routing data in volatile environments such as UAVs. An analysis of the experimental data shows that the integration was successful: the behavior of the FANET was as expected, and the routing protocol behaved adequately in the real-life scenario, ensuring communications during topology changes.

Acknowledgments

This work is funded by FCT/MCTES through national funds under the projects FRIENDS (PTDC/EEI-ROB/28799/2017), S2MovingCity (CMUP-ERI/TIC/0010/2014), and when applicable co-funded EU funds under the project UIDB/50008/2020-UIDP/50008/2020, as well as by the European Regional Development Fund (FEDER), through the Regional Operational Programme of Lisbon (POR LISBOA 2020) and the Competitiveness and Internationalization Operational Programme (COMPETE 2020) of the Portugal 2020 framework under Project 5G (POCI-01-0247-FEDER-024539).

References

[1] "FRIENDS: Fleet of dRones for radIological inspEction, commuNication anD reScue," [Online]. Available: https://www.ipfn.tecnico.ulisboa.pt/FRIENDS/.

[2] D. Camara, "Cavalry to the rescue: Drones fleet to help rescuers operations over disasters scenarios", 2014 IEEE Conference on Antenna Measurements & Applications (CAMA), pp. 1-4. IEEE, 2014.

[3] J. Sanchez-Garcia, J. M. Garcia-Campos, S. L. Toral, D. G. Reina, and F. Barrero, "An intelligent strategy for tactical movements of uavs in disaster scenarios", International Journal of Distributed Sensor Networks 2016 (2016): 18.

[4] H. Chae, J. Park, H. Song, Y. Kim, and H. Jeong, "The iot based automate landing system of a drone for the round-the-clock surveillance solution", 2015 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), 2015.

[5] Donald L. Bryson, Eric V. Kline, and Sarbajit K. Rakshit, "Perimeter monitoring using autonomous drones", U.S. Patent 9,734,684, issued August 15, 2017.

[6] R. Hogg *et al.*, "Algorithms and sensors for small robot path following", IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292), vol. 4, pp. 3850-3857. IEEE, 2002.

[7] MAVLink. [Online]. Available: https://mavlink.io/en/.

[8] Dronekit tutorial. [Online]. Available: http://ardupilot.org/dev/docs/droneapi-tutorial.html.