2022 IEEE 23rd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM) | 978-1-6654-0876-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/WoWMoM54355.2022.00074

Collision-free swarm take-off based on trajectory analysis and UAV grouping

Carles Sastre, Jamie Wubben, Carlos T. Calafate, Juan Carlos Cano and Pietro Manzoni
Department of Computer Engineering (DISCA)
Universitat Politècnica de València, Spain
Email: carsaspe@alumni.upv.es, {jwubben, calafate, jucano, pmanzoni}@disca.upv.es

***Abstract*—In recent years, the adoption of unmanned aerial vehicles (UAVs) has widely spread to different sectors worldwide. Technological advances in this field have made it possible to coordinate the flight of these aircraft so as to conform a swarm. A UAV swarm is defined as a group of UAVs working collaboratively to carry out more complex missions or perform tasks more efficiently. Common applications of these swarms include rescue missions, precision agriculture, and border control, among others. However, there are still certain problems that prevent us from ensuring the success of their mission, especially as the number of drones in a swarm increases. In this paper, we specifically address the problem of a swarm take-off by optimizing the total time involved, while guaranteeing the safety of the UAVs during the take-off stage. To this end, we propose a new approach that combines a collision detection algorithm based on trajectory analysis with a batch generation mechanism that we use in order to determine the take-off sequence. Experiments show that our algorithm offers an efficient solution, managing to improve the performance of existing take-off techniques.**

***Index Terms*—UAV swarm; flight coordination; ArduSim; take-off protocols; safety.**

I. Introduction

Nowadays, the use of Unmanned Aerial Vehicles (UAVs) is progressing very quickly, providing a great number of benefits to our society [1]. The main reasons for such popularity include their low cost, flexibility, and ease of use. Due to the great versatility of these small aircraft, it has become possible to use them together in the form of a swarm, specifically those of the Vertical Take-Off and Landing (VTOL) type. Swarms of UAVs allow us to efficiently perform complex applications that cannot be handled by a single UAV [2]. In the last decade, a large amount of research has been aimed at creating programmed missions for UAV swarms to meet the needs of specific services, especially in the industry. Among the great variability of UAV swarm applications we can find precision agriculture [3], search and rescue missions [4], [5], and carrying heavy loads. The number of UAVs in a swarm is a determining factor in guaranteeing the mission success, as more UAVs increase the redundancy and/or fault tolerance. However, the computational complexity of many algorithms increases as well, since it is often correlated with the number of UAVs. In particular, challenges such as in-flight coordination, communication between the UAVs, handling the loss of swarm elements, and collision avoidance, represent issues that remain mostly untackled by the research community. All these problems prevent swarms of UAVs to experience a widespread adoption.

In this work, we focus on the swarm take-off problem; one of the major challenges yet to be solved. Since during the take-off the UAVs are close to each other, the probability of collision is extremely high. Furthermore, the take-off should be reasonably fast because battery lifetime remains limited. Hence, the traditional sequential approach, despite being safe, is not a feasible approach for large swarms due to its time overhead. Thus, the need for a faster approach arises. We propose a collision-free take-off strategy for UAV swarms that significantly improves take-off times compared to a sequential approach. Our work is composed of two stages. For the first stage, we propose a solution to detect possible collisions between the UAVs trajectories that is both efficient and accurate. In the second stage, we use the information from the first stage to combine UAVs into groups (batches), where within each batch there are no collision risks, thereby enabling a simultaneous takeoff of such UAVs. We tested our approach under various scenarios, using our simulator ArduSim [6]. We measured both the computational overhead and the overall take-off time when varying the number of drones and the formation used. Finally, we compare our approach to the widely used sequential approach, showing that we can achieve reductions on the take-off time of up to 94%, which indeed is a substantial improvement.

The rest of this paper is organized as follows: in Section II an overview of related works is provided. In Section III, the collision detection algorithm is detailed. In Section IV, we detail our proposed batch generation mechanism. Then, in Section V we present an overview of the simulation environment used, along with our experimental results. Finally, in section VI, we present the main conclusions of the work, and discuss future challenges.

978-1-6654-0876-9/22/$31.00 ©2022 IEEE
DOI 10.1109/WoWMoM54355.2022.00074

II. Related work

Currently, research efforts are focused on solving the wide variety of challenges posed by UAV swarms. Not only universities, but also technological companies, are performing a lot of research in this area. In fact, companies like Intel [7] and the Geoscan Group [8] received a lot of attention thanks to their (beautiful) light shows. Unfortunately, very little information has been published detailing these entertainment activities.

Focusing on more traditional research, we can highlight the work by Guanghan Bai et al. [9]. They considered the limitations of the wireless communication between UAVs and, in their work, they propose an improved UAV swarm model. Through this model they were able to create swarms where no nodes were isolated. This improved network connectivity, which leads to more resilience in applications. They showed this improvement using a multi-agent simulation, in which a UAV swarm is called to implement a surveillance mission over a controlled area.

Besides works on resilience, a lot of attention is going towards collision avoidance and swarm formations. In [10] the authors developed a novel algorithm that allows a swarm of UAVs to maintain a formation and avoid collisions. The formation is maintained by constantly assessing the distances between the UAVs, and then slowing down and speeding up individual UAVs whenever necessary. Any obstacles are avoided by detecting the edge of the obstacle and then, with the use of path planning algorithms, flying around the obstacle.

A similar work was published by Jia Wu et al. [11]. In their work, they also presented collision avoidance strategies. Besides avoiding static obstacles, they can also avoid mobile obstacles. They took inspiration from nature (birds and other animals), and created two controllers to integrate flight formation and swarm intelligence into one framework.

The take-off problem, however, has barely been addressed by the research community. This is because, in the past, most swarm applications were using only a few UAVs. In that case, it is still possible to use a sequential take-off procedure. However, nowadays, many applications address larger swarms. Therefore, the take-off problem becomes a prominent issue. Some enhancements have been made by [12]. In that work, a new take-off procedure known as "the semi-sequential" approach was proposed. Although it reduced the take-off time while keeping the risk of collision at zero, it only allowed for two UAVs to take off at the same time. In our work, we are able to take off several UAVs at the same time while still avoiding collisions. This results in an even shorter take-off time.

III. Collision detection

In our proposed approach, we will consider the trajectories of the UAVs, and analyze them for potential collisions before taking off. Since we are able to determine the ground location of each UAV, and the swarm manager defines their corresponding target location (in the air) using the best approach for the specific context [13], their actual trajectories can be known in advance. Our approach can be executed beforehand, or during the take-off process. It is also possible to send the required information to a cloud service, and execute our algorithm on a high-end server. However, in our current approach, we execute our algorithm on the on-board microcomputer (Raspberry Pi) of the master UAV. Since our work is focused on Vertical Take-Off and Landing (VTOL) aircraft, we can assume that the trajectory followed is very close to a straight line. Hence, given that the different trajectories are merely straight lines in a three-dimensional space, we can seamlessly calculate the minimal distance between these trajectories, and thus determine whether two UAVs may collide.

Three different situations can occur: (i) the trajectories are parallel, (ii) they intersect, and (iii) they cross in space (skew). In real world applications, however, the trajectories will almost always be skew lines, as the other cases are statistically unfeasible in real situations.

In the case of skew lines, we can calculate the minimal distance in the following manner. We start by defining lines r and s as:

$$\begin{cases} r = ground1 + (air1 - ground1) \cdot u \\ s = ground2 + (air2 - ground2) \cdot v \end{cases} \tag{1}$$

Line r (and respectively s) can be calculated based on the location of the UAV on the ground, and its desired location in the air. In order to find the shortest distance, we need to find a line which is perpendicular to both lines r and s. This line can be calculated using the cross product:

$$\vec{n} = \vec{r} \times \vec{s} \tag{2}$$

The points where the new line intersect with lines r and s are the points where the distance between r and s is the shortest (i.e. $p1$ on r, and $p2$ on s). These points can be calculated by solving

$$\vec{r} + w \cdot \vec{n} = \vec{s} \tag{3}$$

Once solved, we obtain the values for u, v, and w, which, when filled into 1, determine points $p1$ and $p2$. Finally, the distance between the two points can be calculated using the Pythagoras theorem as:

$$d(p1, p2) = \sqrt{(p1 - p2)^2} \tag{4}$$

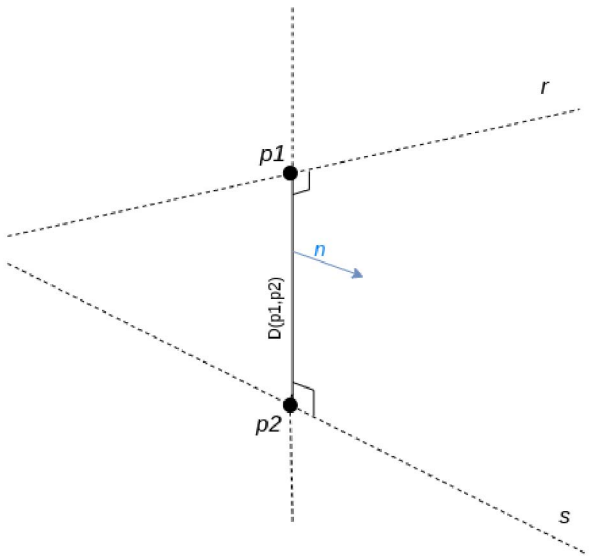

Fig. 1. Line segment representing the minimum distance between two lines.

Given the minimal distance, we can assess whether or not there is a potential danger. During this assessment, we assume that, for both ground and final aerial positions, the distance between UAVs always remains larger than the safety margin (which can be set by the user). Hence, we start by checking if the minimum distance is greater than the safety margin. If so, then there is no conflict. If the minimum distance is smaller than the safety margin, then 3 possible situations can arise:

1) If both points (p1 and p2) are above the ground location, and beneath the swarm air height, we confirm that there is a conflict.
2) If both points (p1 and p2) are either beneath the ground location, or above the air location, we say that there is no danger because the UAVs will never reach such points.
3) If one of the two points (p1 or p2) is within the flight range, but the other is not (either below the initial position, or above the aircraft's final destination), we will replace the latter with the first point that is closest to the actual flight range, i.e. the ground location if the point was underneath it, or the final destination if the point was past it.

Algorithm 1 presents the pseudocode for the collision detection algorithm described above.

IV. Drone batch generation

In Section III, we proposed an algorithm which generates a list of possible collisions. We can now use this information to create batches of UAVs, which do not collide, so that they can take off simultaneously.

Algorithm 1 detectCollision(uavs)

Require: $uncheckedUAV.size = uavs.size$

```
for uavId in uavs do
    uncheckedUAV.remove(uavId)
    for nextUAV in uncheckedUAV do
        r = makeLine(uavId.ground, uavId.air)
        s = makeLine(nextUAV.ground, nextUAV.air)
        if checkParallelVector(r,s) then
            colFlag = false
        else
            colFlag = checkDistMin(r,s)
        end if
        if colFlag then
            collisionList.add(uavId)
            goToLine(1)
        end if
    end for
end for
return collisionList
```

Algorithm 2 batchGe(collisionList,uavs,batchList)

```
list.size = batchList.size
group1.size = uavs.size
group2 = getCollision(collisionList)
group1 = removeDuplicate(group1, group2)
if group2.size > 1 then
    list.add(group1)
    collisionListAux = detectCollision(group2)
    if collisionListAux.size == 0 then
        list.add(group2)
    else
        uavsAux.size = group2.size
        batchGe(collisionListAux,uavsAux,list)
    end if
else
    if !group1.isEmpty() then
        list.add(group1)
    end if
    if !group2.isEmpty() then
        list.add(group2)
    end if
end if
return list
```

Algorithm 2 shows the pseudocode referring to the drone batch generation algorithm we propose.

Three inputs are required for this algorithm to execute. The first one is the list of collisions obtained in the collision detection algorithm. The second one is the list of UAV identifiers. And the third one is the list of previously defined multicopter batches (this list is initially empty); this last element plays a very important role since, being a recursive method, this variable will be in charge of

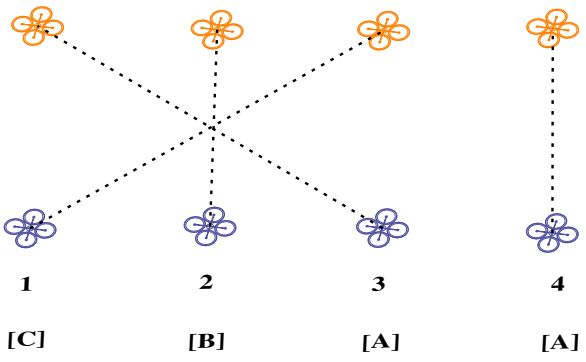

Fig. 2. Example of take-off batch grouping (A,B,C) using our algorithm.

storing the result of the batches defined in the previous iterations of the algorithm.

Our proposal consists of recursively dividing the input list of drones into two groups: one with the drones that do not appear in the collision list (G1), and another with those that do (G2). We add the labels of the UAVs that are part of G1 to the output data as a new batch because they do not generate any conflict between them. Regarding G2, and depending on the number of identifiers in it, there are different situations. If G2 is empty, we will directly return the result of the algorithm. If it has only one element, we will add a new batch with its identifier, and the algorithm will thus end. Finally, if the number of G2 identifiers is greater than or equal to two, we would proceed to create a new list of drones that includes the labels belonging to G2. We will then restart the procedure from the beginning to again determine potential conflicts, except that this time both the list of UAVs and collisions will be different compared to the previous iteration.

In Figure 2 we show an example of application of our batch generation mechanism in a swarm composed of four aircraft. First of all, we observe that the first UAV generates a possible conflict with the second. Therefore, we will add the first drone to the "B" batch, while leaving the second in the "A" group. Then, as a collision is generated between the second and third UAV, we will assign the second UAV to group "B" while the third one remains in group "A". As the third and fourth UAVs do not cross each other's paths, we leave the fourth one in group "A". At this point, we can confirm that UAVs in batch A can take off simultaneously without problems. As the number of drones in group "B" is two, we need to check whether they may collide with each other. Since a conflict persists, we assigned the first drone to batch "C". In this way, we would be able to finish our algorithm by returning the three batches.

V. Experiments and Results

We used the ArduSim simulation environment [6] for the performance evaluation of our proposed solutions. The reason for this choice is that ArduSim has the ability to simulate large numbers of UAVs in real time, and to simulate a FANET for UAV to UAV communications. Regarding UAV positioning, we start from a random ground deployment, as this is the typical situation. Once they have taken off, the swarm must form one of the following regular aerial formations: linear, circular or matrix. Regarding the simulation parameters, we have used a safety distance of 8 meters to account for GPS inaccuracies (below this value a potential collision is detected), and a swarm height of 20 meters. Furthermore, the minimum distance between UAVs on the ground is 10 meters, while in their respective aerial positions it is of 20 meters minimum.

We run the following three experiments:

A. Calculation time of the collision detection algorithm

In this first experiment, we try to analyze the calculation time behavior of our collision detection algorithm. Figure 3 shows the results obtained from the calculation time in the three available aerial formations using different numbers of UAVs. For the matrix formation, we detect collisions with a very low computation time on any number of drones. On the other hand, both in the circular and in the linear formation we see that, beyond 125 UAVs, the calculation time begins to increase significantly. Yet, the overall time is below 10 seconds in all cases, a very reasonable value.

B. Take-off flight time and number of batches generated

In this second experiment, we want to study the take-off flight time (i.e. the time starting from the first UAV taking off until the last UAV reaches its air position), and the number of batches of drones obtained for each of the aerial formations. As we can see in Figure 4, the take-off flight time for the matrix formation increases slightly as we increase the number of aircraft in the swarm, resulting in a time of approximately 3.5 minutes for 150 drones. As far as the linear and circular formations are concerned, they rise much faster than the previous one. In the experiment with 150 drones, we obtain 18.8 and 16.7 minutes for the linear and circular formations, respectively.

To gain insight on how these large values are achieved, Figure 5 shows the number of batches of drones generated. In the matrix formation, we obtain a constant amount of 2 or 3 batches during all the tests carried out, being able to group a greater number of drones in each batch. In the case of the circular formation, we obtain a value of 76 batches for the test with the largest number of drones. Due to the high number of potential collisions found,

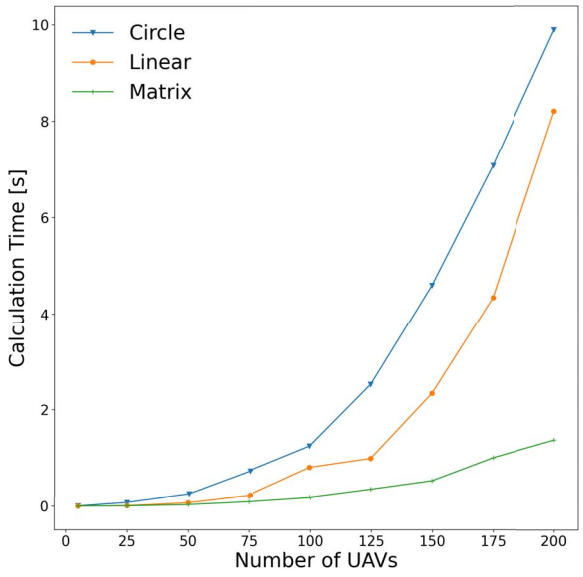

Fig. 3. Calculation time overhead for the collision detection algorithm under different formations.

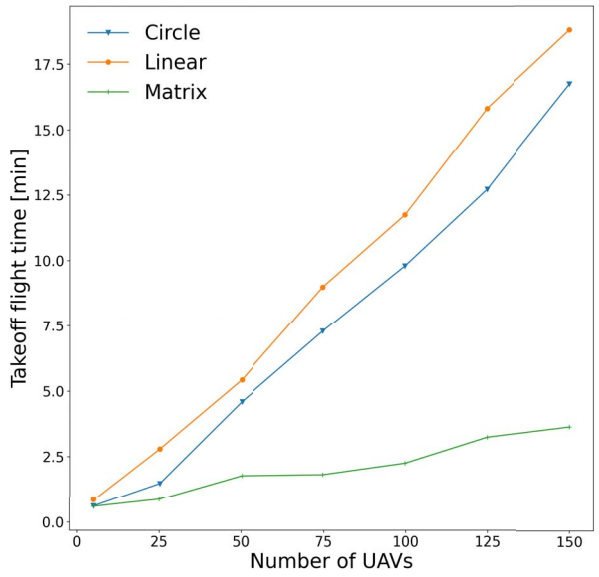

Fig. 4. Take-off flight time w.r.t. different number of UAVs and different formations.

there are some batches having just a single drone. Finally, the linear formation is characterized by the highest travel distances, and by very similar drone trajectories. Thus, similarly to the circular case, the number of conflicts detected is very high, and again there are some batches with just a single drone to guarantee that the risk of collisions is zero.

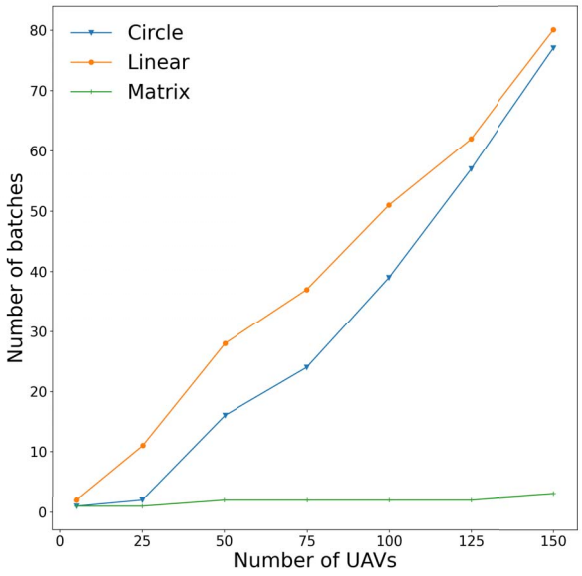

Fig. 5. Number of batches when varying the number of UAVs for different aerial formations.

Another critical factor that explains the high take-off flight time obtained is that we must introduce a wait time (3.5 s) between consecutive batches to make sure that drones belonging to consecutive batches do not collide. This value is used because any lower value proved to generate conflicts, especially for the linear formation. Therefore, both circular and linear formations are the most affected ones due to their high number of batches. Yet, it is worth pointing out that swarms with a high number of drones are usually deployed following a compact formation, like the matrix option in our study, while other formations are less common, and are only applicable to small swarms.

C. Comparison of the take-off flight time against the sequential procedure

In this last experiment, we assess the contribution of our combined algorithms against the standard approach, which is to adopt a sequential take-off strategy.

Figure 6 shows a comparison between our scheme and the sequential one for the available aerial formations. Concerning the linear formation, we managed to reduce the time from almost 4 hours to only 18.8 minutes in the experiment with 150 UAVs. In the case of the circular formation, we reduced it from 2 hours and a half to about 17 minutes. Finally, in the matrix formation, we managed to reduce the take-off time from about 58 minutes to only in 3.6 minutes. In this experiment, we changed the number of UAVs as well as the formation. Since, for

different formations, the total distance travelled is not the same, the sequential take-off flight time for each of them will differ.

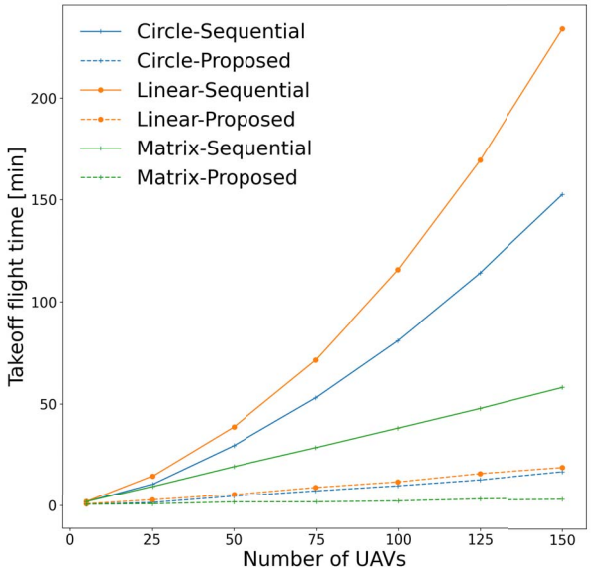

Fig. 6. Comparison with the sequential take-off procedure.

VI. Conclusions and Future Work

Ensuring a fast and collision-free swarm take-off is a basic requirement for any drone swarm application. In fact, considering the short lifetime of UAV batteries, large swarms could be prevented from carrying out more ambitious missions unless such time overhead is reasonably low.

In this work, a new take-off procedure has been proposed that keeps the risk of collision at zero and, at the same time, significantly improves time efficiency. To achieve this, a new collision detection algorithm has been implemented that, combined with the proposed batch generation mechanism, allows us to group the UAVs into batches, where UAVs in each batch can take off simultaneously without causing any conflict. Experimental results show that we are able to reduce take-off time by one order of magnitude compared to the standard sequential approach.

As future work, we plan to further improve performance by reducing the total number of drone batches generated, and by reducing the waiting time between consecutive batches.

Acknowledgments

This work is derived from R&D project RTI2018-096384-B-I00, funded by MCIN/AEI/10.13039/501100011033 and "ERDF A way of making Europe".

References

[1] H. Shakhatreh, A. H. Sawalmeh, A. Al-Fuqaha, Z. Dou, E. Almaita, I. Khalil, N. S. Othman, A. Khreishah, and M. Guizani, "Unmanned aerial vehicles (uavs): A survey on civil applications and key research challenges," *IEEE Access*, vol. 7, pp. 48 572–48 634, 2019.

[2] G. Chmaj and H. Selvaraj, "Distributed processing applications for uav/drones: A survey," in *Progress in Systems Engineering*, H. Selvaraj, D. Zydek, and G. Chmaj, Eds. Cham: Springer International Publishing, 2015, pp. 449–454.

[3] P. Radoglou-Grammatikis, P. Sarigiannidis, T. Lagkas, and I. Moscholios, "A compilation of uav applications for precision agriculture," *Computer Networks*, vol. 172, p. 107148, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S138912862030116X

[4] L. Ruetten, P. A. Regis, D. Feil-Seifer, and S. Sengupta, "Area-optimized uav swarm network for search and rescue operations," in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, 2020, pp. 0613–0618.

[5] R. D. Arnold, H. Yamaguchi, and T. Tanaka, "Extension of a ground control interface for swarms of small drones," *Journal of International Humanitarian Action*, vol. 3, no. 1, 12 2018.

[6] F. Fabra, C. T. Calafate, J. C. Cano, and P. Manzoni, "Ardusim: Accurate and real-time multicopter simulation," *Simulation Modelling Practice and Theory*, vol. 87, pp. 170–190, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1569190X18300893

[7] Intel, "Drone light shows powered by intel," 2022, last visited: 2022-02-07. [Online]. Available: https://www.intel.es/content/www/es/es/technology-innovation/intel-drone-light-shows.html

[8] T. G. Herald, "Thousands of drones light up st. petersburg skies in honor of 75th anniversary of the end of ww2," 2020, last visited: 2022-02-07. [Online]. Available: https://theglobalherald.com/news/thousands-of-drones-light-up-st-petersburg-skies-in-honor-of-75th-anniversary-of-the-end-of-ww2/

[9] G. Bai, Y. Li, Y. Fang, Y.-A. Zhang, and J. Tao, "Network approach for resilience evaluation of a uav swarm by considering communication limits," *Reliability Engineering and System Safety*, vol. 193, p. 106602, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0951832019300171

[10] J. Yasin, M.-H. Haghbayan, J. Heikkonen, H. Tenhunen, and J. Plosila, "Formation maintenance and collision avoidance in a swarm of drones," in *Formation Maintenance and Collision Avoidance in a Swarm of Drones*, 09 2019.

[11] J. Wu, C. Luo, Y. Luo, and K. Li, "Distributed uav swarm formation and collision avoidance strategies over fixed and switching topologies," *IEEE Transactions on Cybernetics*, pp. 1–11, 2021.

[12] F. Fabra, J. Wubben, C. T. Calafate, J. C. Cano, and P. Manzoni, "Efficient and coordinated vertical takeoff of uav swarms," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, 2020, pp. 1–5.

[13] D. Hernández, J. M. Cecília, C. T. Calafate, J.-C. Cano, and P. Manzoni, "The kuhn-munkres algorithm for efficient vertical takeoff of uav swarms," in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*. IEEE, 2021, pp. 1–5.