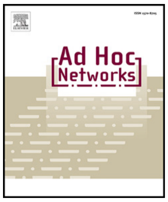
Ad Hoc Networks 140 (2023) 103078

Contents lists available at ScienceDirect

Ad Hoc Networks

journal homepage: www.elsevier.com/locate/adhoc



FFP: A Force Field Protocol for the tactical management of UAV conflicts

Jamie Wubben *, Carlos T. Calafate, Juan-Carlos Cano, Pietro Manzoni

Department of Computer Engineering (DISCA), Universitat Politècnica de València (UPV), Camino de Vera, S/N, 46022, Valencia, Spain

ARTICLE INFO

Keywords:
UAV
Collision avoidance
Artificial potential fields
ArduSim
Tactical management

ABSTRACT

In recent years, we have seen a tremendous growth in the adoption of Unmanned Aerial Vehicles (UAVs). Nowadays, UAVs are used in many different industries such as agriculture, inspection (bridges, pipelines, etc.), parcel delivery, etc. In the near future, this will lead to a substantial increase of aircraft in our airspace, especially in urban areas. Many existing collision avoidance approaches rely on heavy and/or expensive sensors, which limits its use for real UAVs due to increased costs, weight and complexity. Hence, to address this problem, in this paper we present a solution for the tactical management (i.e. in-flight) of UAV conflicts outdoors that introduces minimal requirements: a wireless interface and a GPS module. Specifically, we provide a collision avoidance algorithm based on artificial potential fields to provide flight safety. Our solution, called Force Field Protocol (FFP), allows the UAVs to autonomously detect each other using wireless communications, and to maintain a safe distance between them without the intervention of any central service. Experiments performed in our multi-UAV simulator ArduSim show that, with our approach, collisions between two UAVs are completely avoided in a wide set of scenarios, while introducing low disturbances to the original flight plans. Specifically, in the scenarios that we tested, the additional flight time introduced will be only 7 s longer in the worst case; in addition, it is able to improve upon previous approaches by reducing flight time by up to 54 s. We have shown experimentally that our approach can be scaled easily up to 100 UAVs, and that the probability of a collision is very low (< 0.06) despite flying in a small area (2.5 km × 2.5 km).

1. Introduction

Unmanned Aerial Vehicles (UAVs), colloquial called drones, are becoming an important asset in many industry sectors as they can be used for numerous applications. For instance, they can replace humans in dangerous jobs such as bridge inspection [1]; they can quickly provide first aid [2], especially in areas that are difficult to reach following natural disasters; they can be used to transport organs and blood quickly from one hospital to another during traffic jams [3], and they can increase network connectivity during busy outdoor events. Besides that, UAVs are also used in the film industry [4], precision agriculture [5], border control [6], etc. In summary, UAVs are very versatile, and this characteristic will most definitely lead to an increase in UAVs flying in our airspace. Furthermore, in most applications, we aim for an autonomous flight with minimal or no intervention and monitoring by its owner. All these autonomous UAVs must be able to coherently use a same airspace without crashing into each other and putting our safety at risk.

Current solutions, which are in use with manned aircraft, rely on air traffic management services whereby a control tower is in charge of monitoring and sending orders to the different pilots. However, such an approach will not be feasible because of the high number of UAVs occupying the airspace at one given time, and the low distances involved, which imply near-zero reaction times.

Hence, the need for an ad-hoc approach arises, whereby each UAV takes its own decisions if and when a potential problem is detected. In this work, we present an approach which allows a UAV to avoid other UAVs during flight. Several different types of solutions for this problem exists, which we highlight in the next section. Our approach is based on the concept of artificial potential fields, in which a UAV is attracted by its target waypoint (a GPS location defined by the user), and repelled from any other obstacles. The obstacle might be static or dynamic and, as long as the location is known, it can be nearly anything. However, in this work, we focus specifically on avoiding other (moving) UAVs. To this end, we investigate the influence of various novel repulsion models on various scenarios in our open source multi-UAV simulator, ArduSim [7,8].

Although quite some research has already been conducted towards collision avoidance, there are still some research gaps to bridge. First of all, most of the existing collision avoidance algorithms were created for ground robots. Since UAVs are still relatively new and move in a

---

* Corresponding author.

E-mail addresses: jwubben@disca.upv.es (J. Wubben), calafate@disca.upv.es (C.T. Calafate), jucano@disca.upv.es (J.-C. Cano), pmanzoni@disca.upv.es (P. Manzoni).

https://doi.org/10.1016/j.adhoc.2022.103078
Received 29 March 2022; Received in revised form 7 December 2022; Accepted 21 December 2022

different way (3D mobility, and usually involving much higher speeds) than ground robots, there is still a need to cover the effectiveness of these algorithms on UAVs. Furthermore, a large portion of collisions avoidance algorithms are designed to avoid static obstacles. Naturally, avoiding multiple mobile obstacles is more complicated but, nevertheless, essential for the adoption of UAVs. Finally, most of the current solutions require expensive, large, and/or heavy sensors. These type of solutions might be useful in specific cases but, in general, when we want to fly for as long as possible, we must try to reduce the need of these types of sensors.

Our approach tries to address some of the above-mentioned research gaps. Thus, the main contributions of our work can be summarized as follows:

- Our approach can adapt to both static and dynamic obstacles.
- In terms of hardware requirements, our algorithm only relies on the most basic sensors i.e. a wireless network card and a GPS, which in many cases are already available on the UAV.
- The proposed algorithm introduces a low time overhead compared to the original flight plan.
- We have tested our solution in a realistic UAV simulator, which allows easy deployment on real UAVs.

The rest of this paper is organized as follows: in Section 2, we provide an overview of some related works. In Section 3 we briefly present our multi-UAV simulator ArduSim. In Section 4, we explain in depth how our algorithm works, and detail how the different parameters of our algorithm are optimized. In Section 5 we test our approach in various scenarios, with appropriate discussion. Afterwards, we hold a brief discussion about our work in Section 6. Finally, we conclude our work and provide ideas for future work in Section 7.

2. Related work

As stated above, UAVs can be used in many different industries, which will lead to having multiple concurrent aircraft in the low-level airspace. A major issue is that we are not yet capable of managing this adequately. Traditional manned aircraft use methods such as certification, communication with a control tower, etc. However, due to the number of UAVs, the lack of stability in aircraft design and hardware, and the use of non-traditional aviation-related communications and navigation technologies (e.g. artificial intelligence), these traditional methods are impractical, and not well suited to manage UAVs. Hence, there is a need for an *Unmanned aircraft system Traffic Management (UTM)* [9]. This UTM is intended to be a common framework with core principles for global harmonization. However, it is not intended to propose/endorse specific UTM system design or technical solutions to challenges. Instead, it will provide an overarching framework for such a system. It can be considered as a collection of services that ensure safe and efficient operations of UAVs. One (important) service is the conflict management service, which is subdivided into various services to manage all type of conflicts. Our FFP algorithms falls under the *Dynamic reroute service*, which aims to provide modifications to trajectories to minimize the likelihood of airborne conflicts.

The field of collision avoidance systems (CAS) has been studied for various types of vehicles (ground, water, air) and, given the various existing solutions, a classification of these approaches can and has been made. Such a generalized overview of collision detection and collision avoidance systems is provided in Fig. 1. As stated in [10], each collision avoidance system can be divided into two parts. In the first part (perception) obstacles are detected. The detection of an obstacle is always done by some type of sensor; however, since there are many sensors capable of detecting an obstacle, the authors divide them into two groups: active and passive sensors. The difference stands in the fact that passive sensors measure energy that is naturally available, such as light (e.g., a camera), whereas active sensors require extra energy, which is used to send out a signal that can later be received and interpreted. In Fig. 1 some examples of sensors are given: Sonar, LIDAR, Radar, standard cameras and Infrared cameras. All these sensors allow determining the location of an obstacle in different ways. The choice of the sensors depends highly on the environment. For instance, a camera cannot be used in the dark, but LIDAR can. A popular approach to increase accuracy and built-in resilience is to use various sensors simultaneously. Information from the various sensors is then combined through a technique called sensor fusion. For instance, the authors in [11] explain how LIDAR, and camera data can be fused to obtain better object detection. Data from various sensors can not only be fused, but also transferred. With this technique, explained in [12], data coming from an ultrasonic sound sensor can be converted into data as if it was coming from a LIDAR sensor. This technique proves to be useful because many state-of-the-art methodologies have been designed to work with LIDAR data. However, LIDAR sensors are expensive, and other cheaper sensors, such as ultrasonic sound sensors, can be used if their (sensing) data is properly adapted.

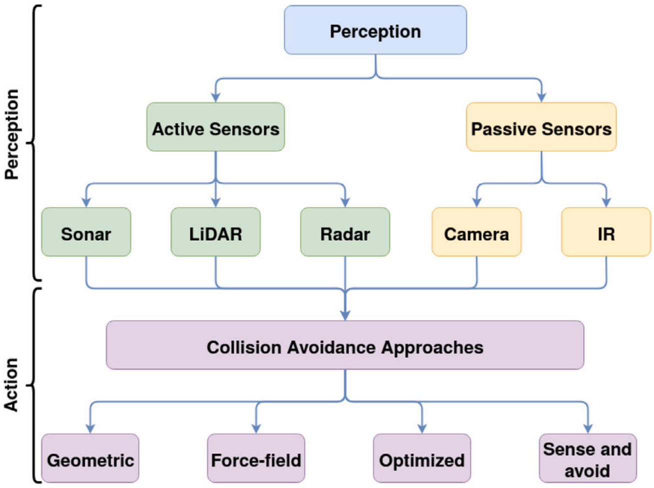


Fig. 1. A general overview of collision perception and collision avoidance systems [10].

In our approach, we decided to broadcast the location of the UAV. In this way, all the UAVs that are in communication range can perceive where the other UAVs are. We have chosen this option because, from September 2023 in the US and January 2024 in the EU all drone manufactures must comply with a new rule about remote identification for unmanned aircraft systems called “remote ID” [13]. Remote ID is the ability of a drone in flight to provide identification and location information that can be received by other parties. Hence, in the future, it becomes feasible for all the UAVs to use our solution.

Once an obstacle is detected, the obstacle still needs to be avoided. The algorithms that avoid collisions can be divided into four main groups. For each group, we provided some works that use an algorithm that falls within that group.

1. **Geometric approaches** rely on the analysis of geometric attributes, such as the distance between the UAVs and their velocities, in order to maintain a safe distance between the UAVs. [14,15]
2. **Force-field approaches** creates an Artificial Potential Field (APF) where, with the use of attraction and repulsion forces, the route of the UAV is altered. Specifically, a UAV is attracted to the place it wants to go, and repelled from any obstacle. The sum of these two vectors determines the new direction vector the UAV should follow. [16–18]
3. **Optimized approaches** rely on calculating a new best route that avoids all (currently known) collisions. These search algorithms are computational expensive, but as the name implies, they

Table 1
Comparative table between various artificial field based approaches.

Author	Mobile/static obstacles	Simulation platform	Directly portable to real UAVs	Obstacle perception
Sun et al. [16]	Static	MATLAB	No	Assumed
Wu et al. [17]	Static	MATLAB	No	Visual image data
Azzabi et al. [25]	Static	MATLAB	No	Assumed
Huang et al. [26]	Static	MATLAB	No	Generic sensor
Choi et al. [27]	Both	Not detailed	No	Assumed
Kownacki et al. [28]	Moving	Numerical	No	Assumed
Ours	Moving	ArduSim	Yes	Communication based

return an optimized result in terms of distance traveled while maintaining a safe distance. [19,20]

4. **Sense-and-avoid** are computational inexpensive methods that react quickly to avoid a nearby obstacle. These approaches are fast because they simplify the process to individual detection and avoidance of obstacles, without taking future plans into account. Often these approaches are used as a last resort. [21,22]

Yasin et al. [10] presents a survey where they explain state-of-art techniques on collision avoidance. Their in-depth review provides a thorough understanding of the various approaches available. Among others, they also explain the Force-field approach. In this paper, we propose an approach that falls into that category. Hence, we compare it with other works that fall in the same category. As shown in Table 1, only a few works exists that are using an artificial field based approach to avoid collision between UAVs. Most of those works are only considering static obstacles, and they are always based on simple simulations. Furthermore, in most of the cases, the position of the obstacle is known *a priori*. Our work differs from the former ones because our approach aims at avoiding multiple collisions in an area with many UAVs. In addition, we make more realistic experiments using a simulator (ArduSim) which is capable of simulating the physical aspects of a UAV with high accuracy, something that cannot be achieved in MATLAB. This allows us to directly deploy the developed protocol in real devices. Finally, we compare the effectiveness of our solution against a geometric protocol available in the literature [23,24] to assess its effectiveness.

3. ArduSim: a multi-UAV simulator

For all the experiments performed in this work, we used a multi-UAV simulator called ArduSim [7]. This simulator is available on GitHub [8] under the Apache License 2.0, and allows us to quickly develop new protocols for handling interactions between UAVs. The most relevant characteristics of ArduSim are detailed below.

- **Fast deployment on real UAVs:** ArduSim is designed to test and run protocols, and later deploy those protocols on real UAVs. To accomplish this, ArduSim is using standards that real UAVs are also using. In this way, simulated code can be transferred quickly and reliably to real UAVs. Deploying a protocol on a real UAV is as simple as connecting a Raspberry Pi to the flight controller using a serial connection. On the Raspberry Pi, ArduSim must be installed, which will send messages to the flight controller using the MAVLink communications protocol [29].
- **Scalability:** As stated before, ArduSim is a multi-UAV simulator. Hence, it is designed to handle a large number of UAVs at once. In essence, the only limitation is the hardware (i.e. the PC), and the protocol used. On a standard workstation (e.g. 32 Gb ram, Intel i7-7700) ArduSim can run up to 500 UAVs.
- **UAV-to-UAV communication:** while using multiple UAVs, simulating the communication between them is essential. Therefore, ArduSim provides four different models. The first model allows all messages to be sent and received. This model is intended to verify if the logic of the protocol does not contain any bugs. The second model drops all packets once a certain distance between the UAVs is surpassed. The third model is based on real experiments between UAVs, and will gradually increase the packet loss w.r.t. the distance between the UAVs. The last model allows the user to connect ArduSim with the OMNeT++ network simulator [30]. Using this co-simulation, the user is free to select the model that better suits the real scenario conditions.
- **API:** Many different protocols need similar functions: taking off, landing, communicating between UAVs, etc. Ardusim provides access to these functions through a straightforward Application Programming Interface (API) in order to achieve faster protocol development.

4. Force Field Protocol (FFP) for collisionless UAV flights

In this section, we describe our proposed contribution, which we denoted as *Force Field Protocol (FFP)*. This protocol is based on the Artificial Potential Fields concept, falling within the Force-field approaches described in Section 2. Next we will present the functional description of this protocol, detailing the different algorithms involved, and afterward we will show how the different protocol parameters have been tuned for optimal operation.

4.1. FFP functional description

The main concept in FFP is that all the aircraft must generate a force field that they share with other nearby aircraft using wireless beaconing (i.e. periodic messages that are broadcasted). In particular, for our solution, all the UAVs broadcast a message with their current location and field strength every 200 ms, a value that provides enough resilience to channel losses while avoiding a high channel occupation. Notice that this communication relies on Wi-Fi ad-hoc communications in the 5 GHz band (802.11ac). In addition to having physical elements (UAVs in our case) that generate a force field between them, which is necessarily of the repulsion type, we also include the possibility to have virtual elements in the scenario that generate force fields as well. Among these we can have fixed obstacles, that again generate repulsion force fields, and target waypoints for the mission of a particular UAV, which generate an attraction force field instead.

Hence, given a specific target location (waypoint) exerting attraction, and the locations of both mobile (UAVs) and static obstacles in the neighborhood, our solution can be applied. Algorithm 1 details the main steps characterizing our proposed FFP.

Algorithm 1 starts by calculating the attraction vector, whose details are presented in Algorithm 2. A vector pointing from the UAV to the target location is created by subtracting the one from the other (i.e. target - UAV). The vector is then reduced to its unit length by normalizing it, and afterward it is scaled based on an attraction function. This attraction function takes into account the distance between the UAV and its target location. As explained later, this will cause the UAV to go at its maximum allowed speed towards the target location during most of its flight. However, in order to prevent that the UAV flies past the target location, its speed must be slightly reduced when the UAV is close to the target.

Algorithm 1 FFP:collisionAvoidance()

```
Require: $targetLocation, locationObstacles$
while !targetReached do
    Vector attraction = getAttractionVector()
    Vector repulsion = getRepulsionVector()
    Vector resulting = Vector.add(attraction, 2*repulsion)
    resulting = resulting.scalarProduct(maxSpeed)
    resulting = reduceToMaxSpeed(resulting)
    moveUAV(resulting)
    if distance(UAV,target) $\leq$ 1 then
        targetReached = true
    end if
end while
```

Algorithm 2 FFP:getAttractionVector()

```
Require: $UAVlocation, targetLocation$
Vector attraction = targetLocation - UAVLocation
attraction.normalize()
$\beta$ = attractionFunction(distance(UAVLocation, targetLocation))
attraction.scalarProduct($\beta$)
return attraction
```

Second, a repulsion vector is calculated. This process (detailed in algorithm 3) is similar to the algorithm for obtaining the attraction vector. However, two major differences exist: (i) in the case of the repulsion vector, the vector points away from the obstacle (otherwise the UAV would fly towards it); and (ii) it is possible that there are multiple obstacles; therefore a loop is used so that each obstacle is taken into account. As in Algorithm 2, a function is used to determine the magnitude of the repulsion vector. Later, we investigate the influence of this repulsion vector. However, in general, the magnitude of the repulsion vector should be large if the obstacle is nearby, and smaller when the obstacle is further away, similarly to the magnetic interaction between same polarity elements.

Algorithm 3 FFP:getRepulsionVector()

```
Require: $UAVlocation, locationObstacles$
Vector totalRepulsion = new Vector();
for obstacle in obstaclesList do
    Vector repulsion = UAVLocation - obstacleLocation
    repulsion.normalize()
    $\gamma$ = repulsionFunction(distance(UAVLocation, obstacleLocation)
    repulsion.scalarProduct($\gamma$)
    totalRepulsion += repulsion
end for
return totalRepulsion
```

The sum of the attraction and two times the repulsion vector determines both the heading and flight speed of the UAV. We multiply the repulsion vector by two because, in the worst case the obstacle and the attraction point are perfectly aligned, and if we simply add the attraction and the repulsion vector the UAV will hover in the air. In order to avoid this, the repulsion vector has to be stronger than the attraction vector. In order to make sure that the UAV will not go faster than the maximum speed that has been set by the user, the length of the resulting vector is reduced in case it exceeds such value. Finally, this resulting vector is passed to the flight controller, which in turn will make sure the UAV flies in the intended direction. This process is repeated until the UAV has reached it target. If the mission consists of multiple waypoints, the target location can simply be replaced by the next one once the previous waypoint has been reached.

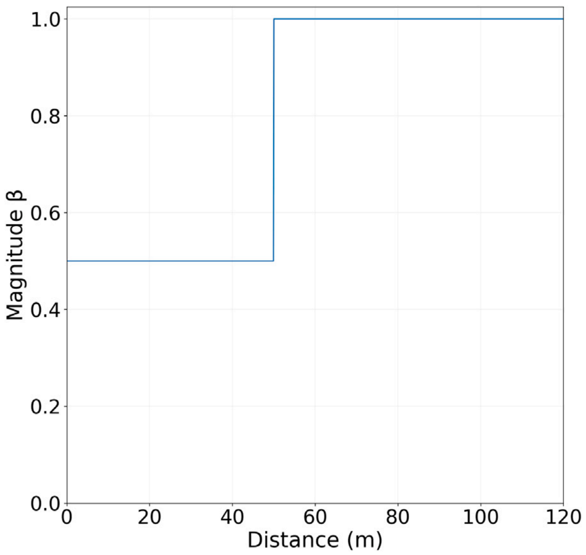


Fig. 2. Attraction function.

4.2. FFP parameter tuning

Now that we have proposed our general approach, we still need to define which functions we are going to use for determining the attraction or repulsion forces between two elements. The chosen functions will have an important impact on the performance of our protocol. Hence, we performed many simulations in order to optimize these functions.

The attraction function is straightforward, since our objective is to go as fast as possible to the target waypoint without surpassing it. Therefore, the magnitude of the attraction function will be equal to its maximum value (i.e. 1). However, we define a threshold when the UAV is within 50 meters to the waypoint where we reduce the magnitude to half (0.5) in order to make sure that the UAV will not surpass the waypoint (deceleration stage); such distance is adequate for UAV deceleration for typical flight speeds (below 20 m/s). In Fig. 2 we provide a graphical representation of this function.

The second and much more interesting function we need to define is the repulsion function. The objective of the repulsion function is to maintain a safe distance between the UAV and the obstacle without introducing a substantial delay to the original flight plan. In general, this means that the function should return a large value when the obstacle is very close. On the contrary, when the obstacle is far away, the function should return zero (or a value close to zero). In order to determine the most adequate repulsion function, we created five different scenarios. These simple missions will serve as benchmarks which allow us to (i) tune the parameters, and (ii) compare our FFP approach with other approaches.

1. **Cross:** in this scenario, two UAVs intersect at 90 degrees.
2. **Angle - same direction:** in this scenario, two UAVs intersect at a 45-degree angle, and both UAVs are going in the same direction, at a same speed.
3. **Angle - opposite direction:** in this scenario, two UAVs intersect at a 45-degree angle, and both UAVs are flying in opposite directions, at the same speed.
4. **Head-on** in this scenario, two UAVs are facing themselves and flying toward each other, at a same speed.
5. **Take-over** in this scenario, one UAV is flying behind another, but it is flying at a faster speed; thus, it will take over the first UAV.

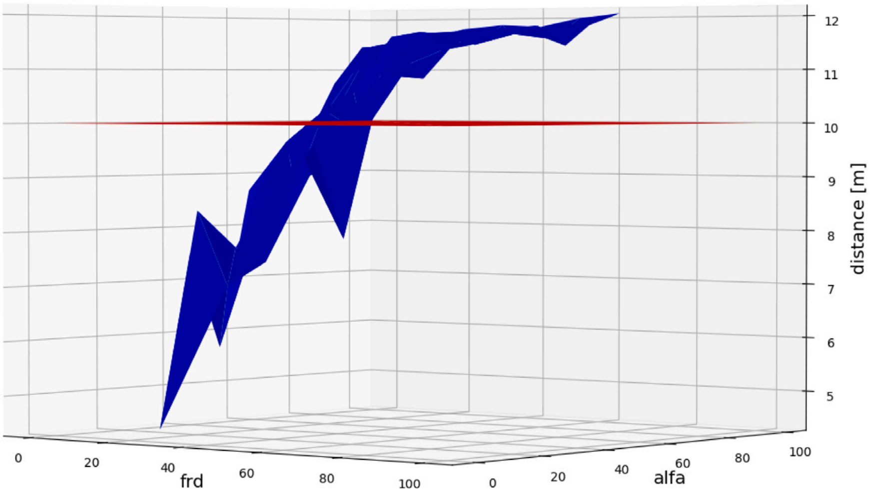


Fig. 3. Minimum distance that the UAV will maintain given a certain frd and α value.

We tested various types of functions in all five scenarios. We first tried families of functions such as $\frac{a}{x-b}$ and, $\log_a(x-b)$, where x is a variable that refers to the distance between the UAV, and the obstacle. Although some functions provided promising results for all five scenarios, we quickly realized that these type of functions are not well suited as a repulsion function due to their asymptotic behavior around b, which prevented the UAV from behaving predictably. In particular, we found that a small change in the distance between the UAV and the obstacle would result in a large difference in the magnitude of the repulsion vector. Notice that small changes could easily come from positioning fluctuations due to GPS error. For that reason, testing the same experiment twice would result in two completely different outcomes. Hence, we decided to discard such functions having an asymptotic behavior.

Next, we experimented with various functions based on a second order polynomials. Specifically, we used the following function

$$Repulsion:\ \gamma = \begin{cases} 1 & \text{if } x \le frd \\ max(1-(\frac{x-frd}{\alpha})^2; 0) & \text{otherwise} \end{cases} \tag{1}$$

frd stands for ”full repulsion distance”, and the α parameter refers to the width of the parabola. Using this function, the UAV will be repelled from the obstacle at maximum strength whenever the obstacle is closer than the full repulsion distance. Further away, the repulsion slowly diminishes according to the chosen α parameter. In all cases, the repulsion vector should be a positive number. In order to find the best second order function, we varied parameter frd between 20 and 80 (with step size 10), and the α parameter between 10 and 80 (with step size 10). Given the five scenarios, this resulted in 280 preliminary tests. In each test, we measured the minimal distance between the UAV and the obstacle, and the extra time it took to move the UAVs to their waypoint.

Fig. 3, shows graphically the results of the above-mentioned experiment. On the z-axis, we show the minimal distance (for all 5 scenarios) given a certain frd and α value. As we can observe, functions with low values for both α and frd are not sufficient to maintain the required safety distance between the UAVs, as illustrated by the plane in red.

Since we also do not want to increase the flight time excessively, we now look further for the function that has the smallest excess flight time (while maintaining a safe distance). To achieve this, we discarded all options that did not maintain a safe distance of 10 meters between the UAV and the obstacle in all five scenarios. Afterwards, we divided the extra time by the minimal distance. This metric allows us to rank the performance of each function. The smaller this number, the better the function, i.e. we prefer a function that introduces the smallest delay while achieving the largest distance between the UAVs.

The set of these solutions are presented in Fig. 4. As we can observe, in this figure there are fewer data points due to removing all the solutions that do not maintain a safe distance at all times. In particular, we can identify the function with $frd = 40$ and $\alpha = 80$ as the best option, since the aforementioned metric is the smallest in this case. The actual shape of this function is depicted in Fig. 5.

5. Simulations & results

In this section, we explain the simulations that we performed to validate our approach, and the performance that we achieve. All the simulations were made in ArduSim [7,8] our real-time UAV emulation platform.

We start by providing a visual representation of how our approach works in the five target scenarios; such output is shown in Fig. 6. In red, we can see the repulsion vector that is generated at any given time. The attraction vector will always point towards the target waypoint, and therefore we have omitted it from the figures. As we can see, the magnitude of the repulsion vector grows when the UAVs are closer to each other. This way, whenever the UAVs get too close, the repulsion magnitude is enough for the UAVs to repel each other. Hence, the minimum distance is always maintained, meaning that collisions are avoided. As clearly shown in Fig. 6(b), when the UAVs are flying in similar directions, some “oscillation” takes place. This behavior is not optimal, and it occurs because we do not take the direction from where the mobile obstacle (other UAV) is coming from into account. Nevertheless, at all times the safe distance is maintained, and eventually the UAVs reach their waypoint. During these simulations, we measured the minimal distance between the two UAVs, the energy usage, and the time it took for the UAVs to reach the different waypoints. In Table 2 we compare our solution to the same five scenarios without using any collision avoidance protocol. The goal of this experiment is to measure the impact of changing the flight plan in terms of flight time, energy usage, and minimal distance.

5.1. Comparison against a geometric protocol

To better assess the performance achieved by our solution, we will compare results of the simulations described above against an alternative approach known as MBCAP (Mission Based Collision Avoidance Protocol for UAVs) that was presented in [23,24], and that falls under the geometric category. The full explanation on how MBCAP works is outside the scope for this work. However, we will highlight the main differences between MBCAP, and FFP. In short, MBCAP works by broadcasting messages between the UAVs, and these messages contain

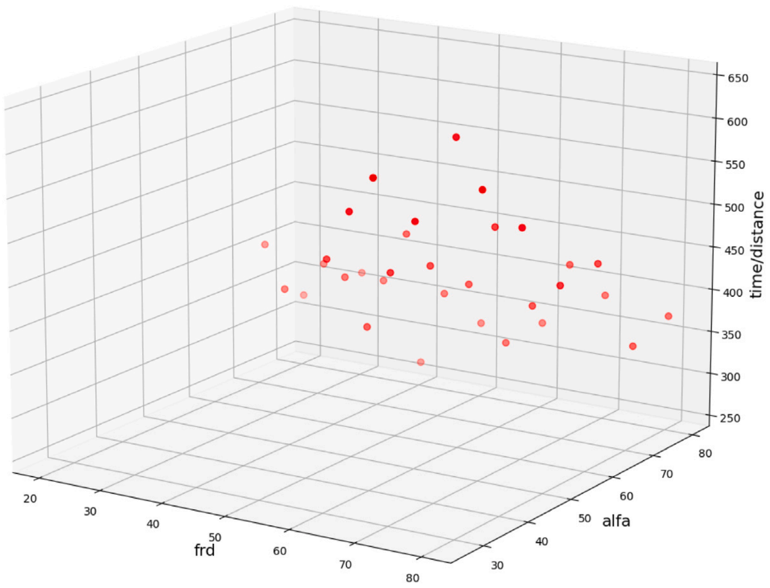


Fig. 4. Excess time/minimal distance given a certain frd and α value.

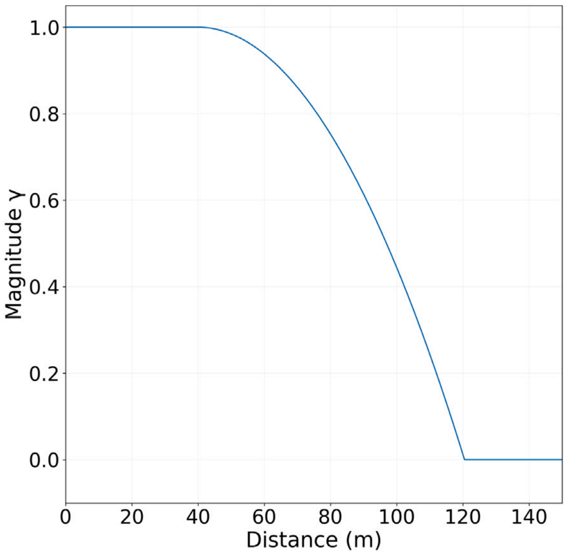


Fig. 5. Repulsion function.

Table 2
The time, energy, and safety for various scenarios compared to no collision avoidance algorithm.

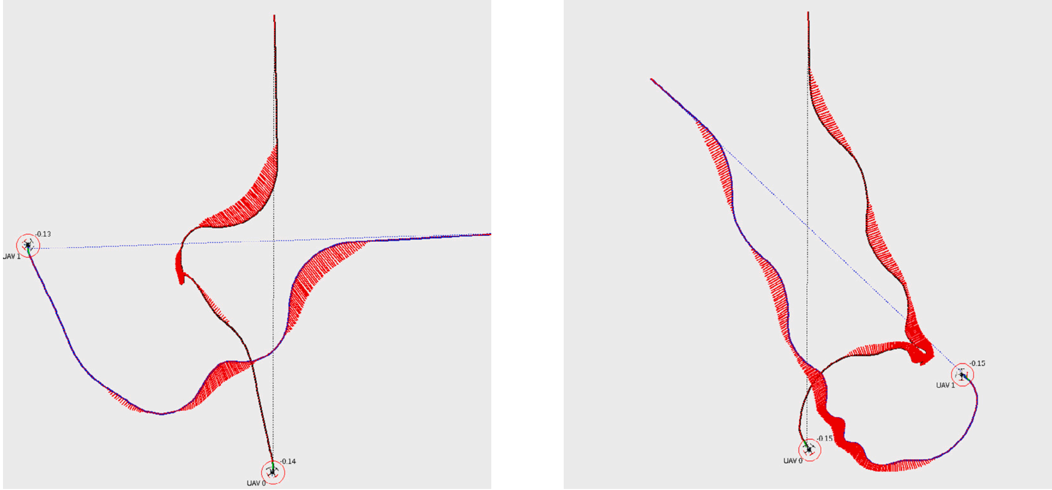
Scenario	1	2	3	4	5	Average
Time [s]						
Without FFP	47	41	46	46	173	
With FFP	76	80	71	60	178	
Δ	29	39	25	14	5	22.4
Energy [mAh]						
Without FFP	570	570	570	570	1575	
With FFP	825	848	780	690	1725	
Δ	255	278	210	20	150	182.6
Min. distance[m]						
Without FFP	0	0	0	0	0	
With FFP	62	91	24	18	11	
Δ	62	91	24	18	11	41.2

the positions the UAV will occupy in the next 25 s. Given this array of positions, the other UAVs can calculate if there will be a collision in the near future. In case a collision risk is detected, the UAVs will rely on a finite state machine to handle it. Each UAV has a priority number, and the UAV with the lowest priority number is required to move aside, so that the UAV with higher priority can continue its flight. Once the collision is avoided, the UAV that moved aside goes back to its original flight path, continuing its flight.

In general, both MBCAP and FFP can be considered equally valid approaches to the tactical handling of aerial conflicts during flight. The main differences between MBCAP and the approach that we propose in this work (FFP) are the following:

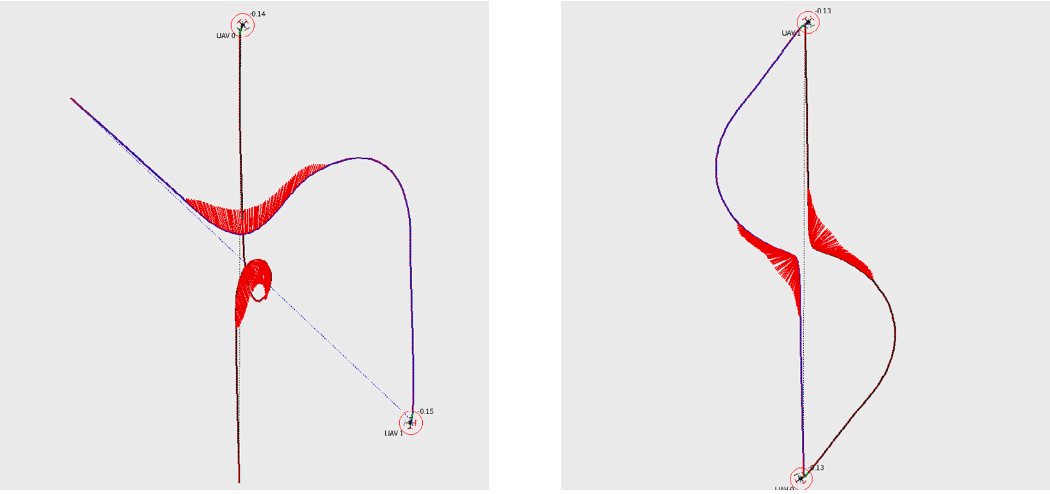
1. MBCAP is deterministic; this is not necessarily bad, but it does imply that a large finite state machine has to be created. This finite state machine contains many states, with some exceptions for special cases (such as deadlocks). The approach we propose in this work is much more general, elegant, and simple, while it can be understood more easily.
2. To avoid a collision, MBCAP may at times require one of the UAVs to move aside to allow enough room for the other UAV to pass by under adequate safety conditions. In the current approach, we move both UAVs, and they keep on flying. This might result in a shorter conflict resolution time.
3. MBCAP only works for dynamic (i.e. moving) obstacles, whereas our current approach can also work with static obstacles.
4. MBCAP mostly preserves the original flight path that was planned, with minor deviations. Our current proposal, however, can introduce more deviations from the intended flight path in certain cases.

In Table 3 we present the results obtained. As explained above, the time columns refer to the excess time it took the UAVs to reach the waypoints, i.e. the difference towards a straight flight where no collision avoidance strategy is used. Furthermore, since we alter the route, it might take one UAV longer to finish the flight. In all cases, we measured the time until both UAVs finished their flight (worst-case values). As we can observe from Table 3, both approaches ensure that the minimal distance is maintained. However, in most cases, our proposed solution (FFP) introduces a larger safety distance between both UAVs, while at the same time reduces the total flight time. On average, our current approach decreases the additional flight time by



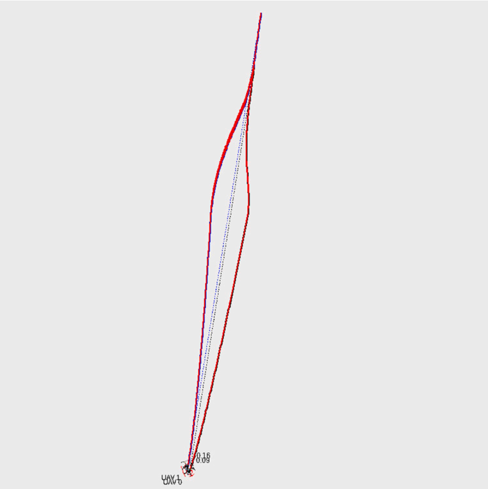
(a) Scenario 1: cross.

(b) Scenario 2: Angle/same direction.



(c) Scenario 3: Angle/opposite direction.

(d) Scenario 4: Head on.



(e) Scenario 5: take over.

Fig. 6. Visual representation of the various scenarios being tested. In red the repulsion vector is shown.

41%, while increasing the minimal distance between UAVs by 62% on average. These significant improvements are achieved because, in our proposed solution, the UAVs keep moving instead of halting, and most of the time they keep reducing their distance to the target waypoint. It is worth mentioning the very significant flight time improvements for scenarios 4 and 5, and especially for the latter one, where a conflict that previously required 59 additional seconds to be solved, now can be solved with just 5 additional flight time seconds. This represents a 91.5% decrease, and evidences the high effectiveness of our proposed FFP method in specific situations.

Table 3
Differences between our proposed solution (FFP) and MBCAP [23].

Scenario	Excess time [s]			Min. distance [m]		
	FFP	MBCAP	Δ	FFP	MBCAP	Δ
1	29	22	+7	62	29	+33
2	39	30	+9	91	35	+56
3	25	35	−10	24	27	−3
4	14	45	−31	18	16	−2
5	5	59	−54	11	20	−9
Mean	22.4	38.2	−15.8	41.2	25.4	+15

Furthermore, we also captured the distance towards the waypoint at every time instance. For these graphs, we always used the flight path of UAV 1 to ensure that the start and the end position will be exactly the same. As we can observe from Fig. 7, one of the advantages of our FFP approach (which reduces the flight time) is that the UAV keeps moving, while for MBCAP there are stalled periods. Furthermore, we can also observe that, in most cases, the UAV keeps flying towards the waypoint. Only for scenario 3 (see Fig. 7(c)) is the UAV required to move a bit away from the waypoint in order to avoid the collision. This maneuver (which can also be observed in Fig. 6(c)) took around 20 s; this is also the reason why the overall flight time (in this specific scenario) of the FFP approach was longer than for the MBCAP approach.

5.2. Influence of beaconing frequency

In all of our experiments, the UAVs are broadcasting their position every 200 ms. However, the frequency of which the UAV broadcast their position (i.e. the beaconing frequency) might influence the performance of our algorithm. The faster the UAVs broadcast their position, the more accurate the information. However, broadcasting very often does saturate the network. In order to assess the influence of the beaconing frequency, we perform various experiments with our simulator. For each scenario, we perform multiple test, each time varying the beaconing frequency. We range the time between sending the beacons from 200 ms (i.e. 5 Hz) up to 2000 ms (i.e. 0.5 Hz) with a step size of 200 ms. During those tests, we measure the minimal distance between the two UAVs. The results are shown in Fig. 8. As expected, the minimal distances between the UAVs does decrease when the interval time increases. Resulting in a higher chance of collision. However, this effect is minimal, and our robust algorithm is still able to maintain a safe distance between the UAVs even when the time between broadcasting the beacons grows.

5.3. Influence of difference in UAV speeds

In our other experiments, we assume that all the UAVs are flying at the same speed (10 m/s). This is of course not something that we can assume for real world applications. Hence, in this experiment, we vary the speed of the UAVs. For this experiment we only use scenario 4 and 5 because in the other scenarios the UAVs would no longer collide. One UAV is consistently flying at 5 m/s while the other UAV's speed changes for each experiment. we range the speed of the second UAV between 8 m/s and 15 m/s (i.e. a difference of 3 m/s up to 10 m/s). As shown in Fig. 9, in both scenarios, the minimal distance between the two UAVs stays consistent. Again, showing that our algorithm is robust and is not effected by the difference in UAV speeds.

5.4. Scalability analysis

Now that we have validated the effectiveness of our approach in five different representative scenarios, and the influence of several parameters, we want to test its scalability. To that end, we performed various simulations where UAVs are flying following a random path (from one point to another). For each experiment, we will increase the number of UAVs (starting from 10, and up to 100, with a stepsize of 10). During the simulations, the UAVs are initially placed at a random position within a square of size 2.5 × 2.5 km (with at least 15 meters between each UAV). They then follow a random (straight line) path, which is at least 200 meters long. An example of such a flight is given in Fig. 10. The UAV's maximum speed is 10 m/s, although occasionally this speed might be lower (when it is avoiding an obstacle). The UAVs broadcast their position using UDP messaging over Wi-Fi (802.11n) in the 5.8 GHz band. In ArduSim, this communication is modeled based on theory and real experiments. Hence, UAVs will only (probabilistically) receive the messages when they are within range, and there is no network collision. Results are shown in Table 4.

Table 4
Nr. of packets send and received during the mission.

Nr. UAVs	Total number of packets sent	Potentially received packet	Not received	
			Out of range	Channel collision
10	5669	51021	6646 (13%)	81 (0.16%)
20	9378	178182	33671 (18%)	85 (0.05%)
30	21295	617555	99094 (16%)	191 (0.03%)
40	23694	924066	168255 (18%)	184 (0.02%)

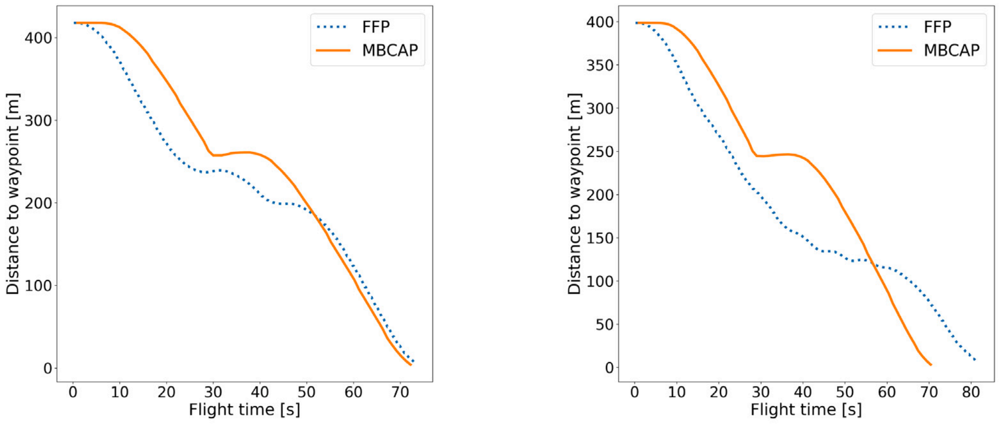
Table 5
Characterization of the minimum distance distribution and chances of collision (i.e. distance < 10 m) when varying the total number of UAVs.

Nr. of UAVs	Without FFP			With FFP		
	μ [m]	σ [m]	P(0 < x < 10)	μ [m]	σ [m]	P(0 < x < 10)
10	143	119	0.13	155	112	0.01
20	72	87	0.24	91	80	0.03
30	55	64	0.24	74	58	0.03
40	47	57	0.25	64	52	0.04
50	34	39	0.26	55	36	0.04
60	29	34	0.29	51	36	0.05
70	26	31	0.29	49	33	0.05
80	25	25	0.28	48	29	0.05
90	22	23	0.29	46	27	0.05
100	20	22	0.32	41	30	0.06

In terms of performance, our purpose is to retrieve more data that can provide further insight into the actual effectiveness of FFP in challenged environments (high air traffic congestion scenarios). To this end, instead of just measuring a single overall value for the minimal distance between UAVs, we measure the minimal distance that is experienced by each UAV. In particular, we ran our simulations several times (changing the initial locations randomly), so that we have at least 100 data points. These data allow us to calculate the mean and standard deviation of the minimal distance values. Fig. 11 shows the normal distributions that describes the set of values obtained on the different simulations.

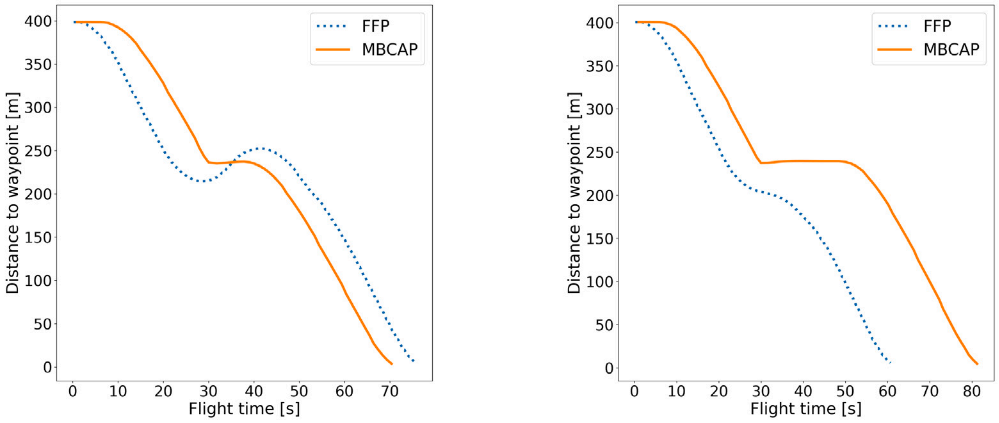
As we can observe in Fig. 11, the minimal distance detected between two UAVs decreases when the total number of UAVs (i.e. obstacles) increases. Evidently this is expected since the squared area where the UAVs are flying is maintained the same (i.e. 2.5 km × 2.5 km). Hence, when there are more UAVs in the same area, the node density increases, and so the minimum detected distance between UAVs is expected to decrease. However, when using our approach, the UAVs should not come too close (i.e. possible collisions should be avoided). Based on this normal distribution, we can easily determine the chances of potential collisions to take place. In Table 5 we present this information. Due to the GPS error, the actual UAV position at a certain time could slightly differ from the one which is assumed, and so this error has to be taking into account when we want to determine the chances of collision. In general, the GPS-error is estimated to be of 5 m [31]. Since this error occurs on both UAVs, and since it can be accumulative, we consider that any distance smaller than 10 meters is a potential collision. In Table 5 we have calculated the chances of that event happening depending on the number of UAVs in the scenario.

The results shown evidence that FFP promotes great overall distances between UAVs, no matter how many UAVs can be found in



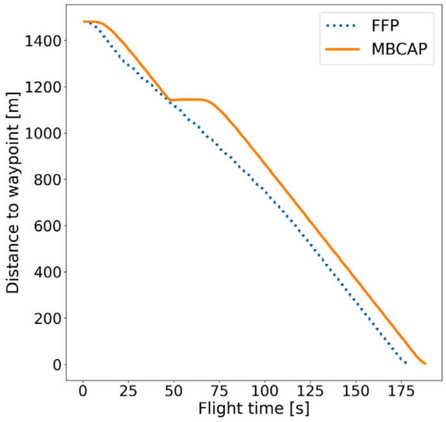
(a) Scenario 1: cross.

(b) Scenario 2: Angle/same direction.



(c) Scenario 3: Angle/opposite direction.

(d) Scenario 4: Head on.



(e) Scenario 5: take over.

Fig. 7. Distance to the waypoint for the various scenarios.

the scenario. In fact, even when the density is very high (100 UAVs), we find that the minimum distance between UAVs in on average 41 m, about twice the value we have when FFP is not used. If we focus specifically on potential collisions (i.e. distance between UAVs below 10 m), we observe how the chances for this to occur decrease very substantially when using our FFP approach. Normally, when no collision avoidance algorithm is used, the chances are between 13% for 10 UAVs, and up to 32% for 100 UAVs. Whereas, by using our FFP approach, we can see that such collision chances are always maintained below 6%. Such value, although still having margin for improvement, represents a good performance level that is similar to other proposals found in the literature [24].

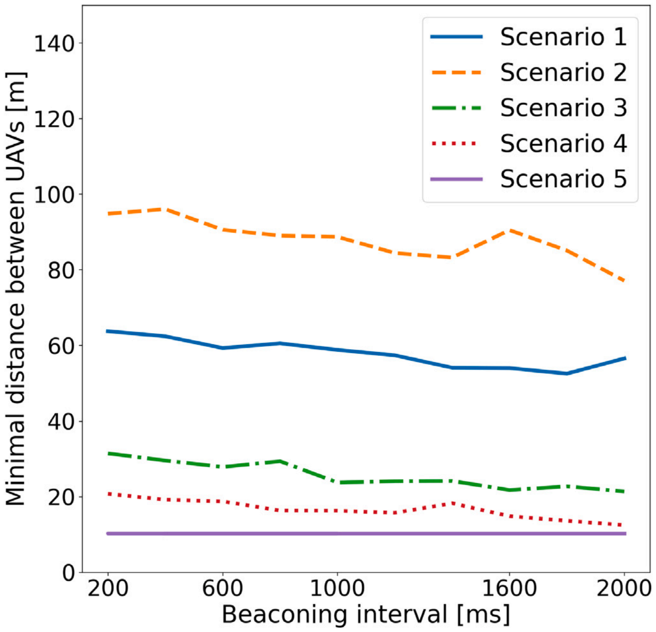


Fig. 8. Minimal distance for each scenario while changing the beaconing interval.

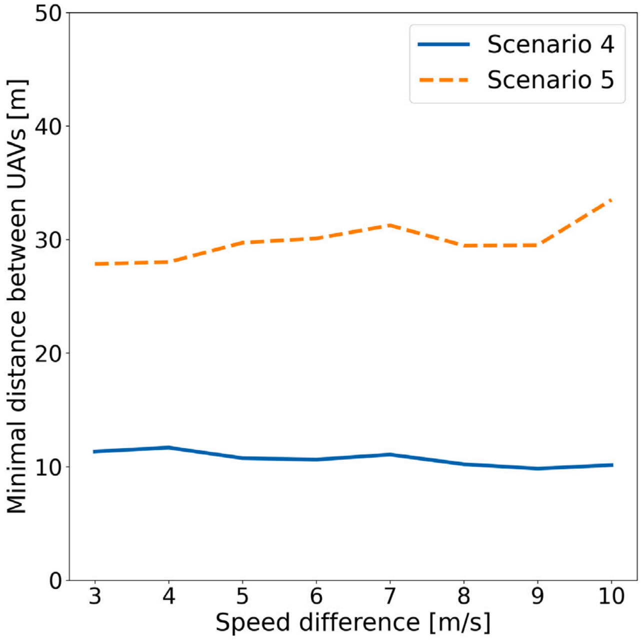


Fig. 9. Influence of difference in UAV speeds.

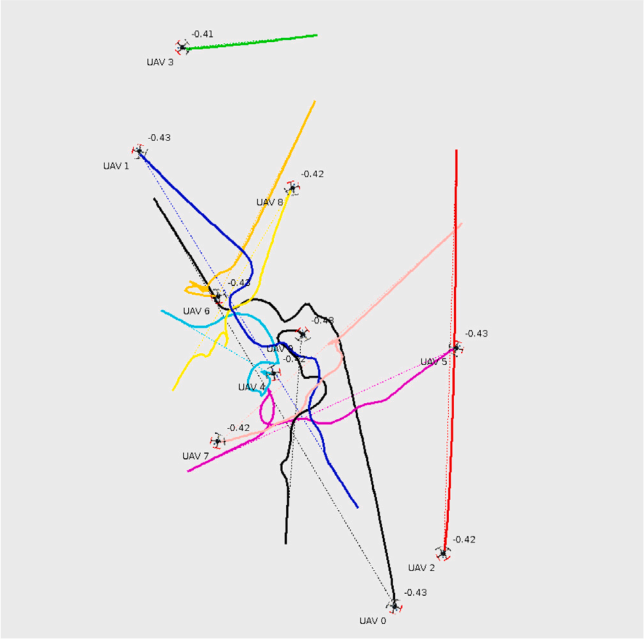


Fig. 10. Example of 10 UAVs performing their random mission.

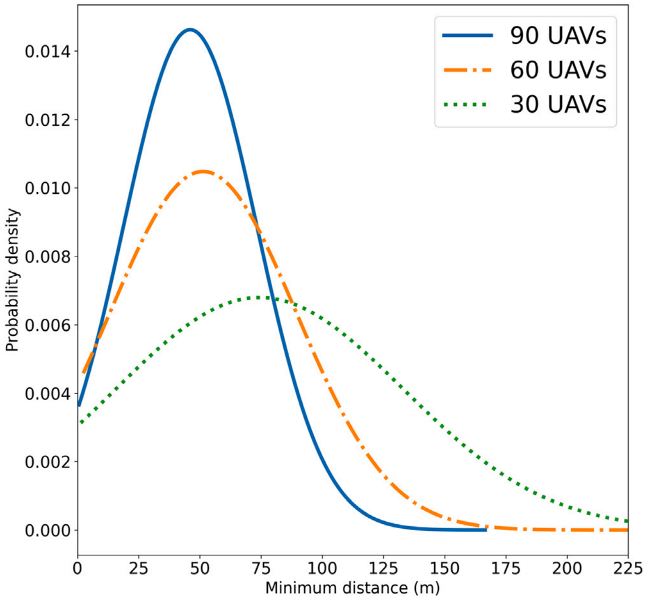


Fig. 11. Normal distribution for the minimum distance values obtained when testing scenarios with different numbers of UAVs.

6. Discussion

In this section, we discuss several issues and lessons learnt from our work. We will start by explaining the strengths and weaknesses of our FFP algorithm. Then we compare it to our previous algorithm MBCAP, and explain in which scenarios one algorithm can be preferred over another. Finally, we discuss other wireless technologies that could be used instead of Wi-Fi, and what impact this might have on the performance of our FFP algorithm.

Focusing first on the strengths of the FFP protocol, we find that the algorithms are computational inexpensive, they scale well, and can be calculated quickly even when there are many obstacles. In addition, the repulsion vector can easily be adjusted so that a larger distance between the UAVs is maintained if required. Moreover, our approach can adapt to both static and mobile obstacles. Finally, our solution allows the UAVs to keep moving, which tends to decrease the overall flight time (w.r.t. other approaches that stop the UAVs until an issue is resolved), meaning also that integration with fixed-wing UAVs can be considered. However, the main weakness of our approach is that the movement of the UAV cannot easily be predicted without running the simulation experiment. In particular, when there are multiple obstacles present, it is difficult to gain an intuitive understanding of where the UAV will go next as it is dynamically adapting its flight according to its perceptive conditions.

These strengths and weaknesses will eventually determine in which scenarios our FFP algorithm is best used. As explained above, the FFP

Table 6
Comparison between various communication technologies.

Communication technology	IEEE standard	Theoretical data rate	Outdoor range
Wi-Fi	802.11n	600 Mbps	250 m
	802.11p	27 Mbps	1000 m
Bluetooth 5	802.15.1	2 Mbps	200 m
ZigBee	802.15.4	250 Kbps	100 m
Wireless Hart	802.15.4	250 Kbps	250 m
Sigfox	N/A	50 kbps	40 km

algorithm will always keep the UAVs moving. This leads (in general) to a small time overhead, which in many cases is preferred. However, it will cause the UAVs to deviate from their pre-planned mission, and this behavior is difficult to predict beforehand. Hence, we believe that this algorithm is best used in scenarios when UAVs can move freely through the 3D space. However, when slim air corridors are used, minor deviations to the flight plan and more predictable movements are preferred. In those cases, the user might opt for our deterministic algorithm MBCAP instead.

For our approach, we have used Wi-Fi technology to establish communications between the UAVs. We mainly use Wi-Fi for the following four reasons: (i) many electronic devices have the hardware required to communicate using Wi-Fi; (ii) the technology exists for a long time (first generation in 1997), and it is actively maintained; (iii) it allows modules to communicate in an ad-hoc mode, and thus does not necessarily rely on any infrastructure to work; and (iv) it allows two modules to communicate over long distances with a relative fast speed.

Although, Wi-Fi technology might seem the most straightforward approach to communicate between UAVs, it is not the only option available. In cases where the UAVs also need to communicate with devices on the ground, the topic becomes very interesting, and it is actively investigated [32]. The main issue is that ground antennas, available as part of the cellular infrastructure, are not designed to provide coverage in the skies.

The main criteria for picking a specific wireless technology are communication range, power consumption, antenna size and speed. Assessing the communication range is complicated because many experiments were conducted on the ground, where there are more obstacles and other phenomena that interfere with the signal. Nevertheless, we give an idea of the expected range in Table 6 along with the theoretical data rate. As one can see, although there exist many options, Wi-Fi is the most suitable because it provides a good trade-off between communication range and data rate. Although the power consumption of Wi-Fi is not the best among these technologies, reaching up to 3 W, it remains acceptable for UAVs. In terms of antenna size, all these technologies are associated to high frequency bands, meaning that UAVs are able to carry them.

7. Conclusions and future work

The increased adoption of UAVs for an ever-growing number of tasks must be accompanied by methods that are able to guarantee flight safety, especially in those environments where the density of UAVs flying simultaneously is expected to be high. To this end, different techniques have been devised in the past few years, adopting a variety of approaches.

In this work we presented FFP, a protocol based on the Artificial Force Field concept that is able to offer good performance by reducing the time overhead introduced by collision avoidance procedures, while maintaining safety distances high. Through different simulation experiments we have shown that the combination of attraction and repulsion concepts adopted by our protocol is able to adequately manage the positioning error introduced by GPS-based localization, and to reduce the additional flight time of UAVs by up to 91.5% compared to another state-of-the-art solution (MBCAP). Also, in very dense environments, it is able to substantially improve flight safety, increasing the minimum distance towards other UAVs between 12 and 25 m.

As future work, we plan to introduce directional force fields instead of omnidirectional ones in an attempt to further boost performance by reducing the time overhead involved in the different maneuvers. Furthermore, we plan to compare our current FFP approach with other approaches such as machine learning, optimized, and sense and avoid based approaches.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work is derived from R&D project PID2021-122580NB-I00, funded by MCIN/AEI/10.13039/501100011033 and “ERDF A way of making Europe.

References

[1] G. Delepine, Drone autonomy for bridge inspection. The newest tool on the inspector’s belt, 2020, https://medium.com/skydio/drone-autonomy-for-bridge-inspection-the-newest-tool-on-the-inspectors-belt-68be85065b13. (Accessed 15 April 2022).

[2] S.S. Fakhrulddin, S.K. Gharghan, A. Al-Naji, J. Chahl, An advanced first aid system based on an unmanned aerial vehicles and a Wireless Body Area sensor network for elderly persons in outdoor environments, Sensors 19 (13) (2019) http://dx.doi.org/10.3390/s19132955, URL https://www.mdpi.com/1424-8220/19/13/2955.

[3] J. Scalea, T. Pucciarella, T. Talaie, S. Restaino, C. Drachenberg, C. Alexander, T. Qaoud, R. Barth, N. Wereley, M. Scassero, Successful implementation of unmanned aircraft use for delivery of a human organ for transplantation, Ann. Surg. (2019) http://dx.doi.org/10.1097/SLA.0000000000003630, Publish Ahead of Print.

[4] R. Gilbey, Eye in the sky: how drone technology is transforming film-making, 2020, https://www.theguardian.com/film/2020/aug/31/how-drone-technology-is-transforming-film-making-french-drama-les-miserables. (Accessed 15 April 2022).

[5] D. Liuzza, G. Silano, F. Picariello, L. Iannelli, L. Glielmo, L. De Vito, P. Daponte, A review on the use of drones for precision agriculture, IOP Conf. Ser.: Earth Environ. Sci. 275 (2018) http://dx.doi.org/10.1088/1755-1315/275/1/012022.

[6] R. Koslowski, M. Schulzke, Drones along borders: Border security UAVs in the United States and the European Union, Int. Stud. Perspect. 19 (2017) http://dx.doi.org/10.1093/isp/eky002.

[7] F. Fabra, C.T. Calafate, J.C. Cano, P. Manzoni, Ardusim: Accurate and real-time multicopter simulation, Simul. Model. Pract. Theory 87 (2018) 170–190, http://dx.doi.org/10.1016/j.simpat.2018.06.009, URL https://www.sciencedirect.com/science/article/pii/S1569190X18300893.

[8] GRCDev, ArduSim: Accurate and real-time multi-UAV simulation, 2020, https://github.com/GRCDEV/ArduSim. (Accessed 15 April 2022).

[9] International Civil Aviation Organization, Unmanned aircraft systems traffic management (UTM) – A common framework with core principles for global harmonization edition 3, 2021, https://www.icao.int/safety/UA/Documents/UTM%20Framework%20Edition%203.pdf.

[10] J.N. Yasin, S.A.S. Mohamed, M.-H. Haghbayan, J. Heikkonen, H. Tenhunen, J. Plosila, Unmanned Aerial Vehicles (UAVs): Collision avoidance systems and approaches, IEEE Access 8 (2020) 105139–105155, http://dx.doi.org/10.1109/ACCESS.2020.3000064.

[11] D. Balemans, S. Vanneste, J. de Hoog, S. Mercelis, P. Hellinckx, Lidar and camera sensor fusion for 2D and 3D object detection, in: L. Barolli, P. Hellinckx, J. Natwichai (Eds.), Advances on P2P, Parallel, Grid, Cloud and Internet Computing, Springer International Publishing, Cham, 2020, pp. 798–807.

[12] N. Balemans, P. Hellinckx, J. Steckel, Predicting LiDAR data from sonar images, IEEE Access 9 (2021) 57897–57906, http://dx.doi.org/10.1109/ACCESS.2021.3072551.

[13] Federal aviation Administration, UAS remote identifaction, 2022, https://www.faa.gov/uas/getting_started/remote_id. (Accessed 18 November 2022).

[14] J.-W. Park, H. Oh, M.-J. Tahk, UAV collision avoidance based on geometric approach, in: 2008 SICE Annual Conference, Loughborough University, Loughborough, England, 2008, pp. 2122–2126, http://dx.doi.org/10.1109/SICE.2008.4655013.

[15] K. Bilimoria, A geometric optimization approach to aircraft conflict resolution, 2000, http://dx.doi.org/10.2514/6.2000-4265.

[16] J. Sun, J. Tang, S. Lao, Collision avoidance for cooperative UAVs with optimized artificial potential field algorithm, IEEE Access 5 (2017) 18382–18390, http://dx.doi.org/10.1109/ACCESS.2017.2746752.

[17] E. Wu, Y. Sun, J. Huang, C. Zhang, Z. Li, Multi UAV cluster control method based on virtual core in improved artificial potential field, IEEE Access 8 (2020) 131647–131661, http://dx.doi.org/10.1109/ACCESS.2020.3009972.

[18] Y. Liu, Y. Zhao, A virtual-waypoint based artificial potential field method for UAV path planning, in: CGNCC 2016 - 2016 IEEE Chinese Guidance, Navigation and Control Conference, Institute of Electrical and Electronics Engineers Inc., Manhattan, New York, U.S, 2017, pp. 949–953, http://dx.doi.org/10.1109/CGNCC.2016.7828913.

[19] B.M. Albaker, N.A. Rahim, A survey of collision avoidance approaches for unmanned aerial vehicles, in: 2009 International Conference for Technical Postgraduates, TECHPOS, IEEE, Manhattan, New York, U.S, 2009, pp. 1–7, http://dx.doi.org/10.1109/TECHPOS.2009.5412074.

[20] S. Pérez-Carabaza, J. Scherer, B. Rinner, J.A. López-Orozco, E. Besada-Portas, UAV trajectory optimization for minimum time search with communication constraints and collision avoidance, Eng. Appl. Artif. Intell. 85 (2019) 357–371, http://dx.doi.org/10.1016/j.engappai.2019.06.002.

[21] W. Min, V. Holger, S. Daobilige, Robust online obstacle detection and tracking for collision-freenavigation of multirotor UAVs in complex environments, in: 2018 15th International Conference on Control, Automation, Robotics and Vision, ICARCV, IEEE, 2018, pp. 1228–1234, http://dx.doi.org/10.1109/ICARCV.2018.8581330.

[22] J.N. Yasin, M.-H. Haghbayan, J. Heikkonen, H. Tenhunen, J. Plosila, Formation maintenance and collision avoidance in a swarm of drones, in: ISCSIC 2019, Association for Computing Machinery, New York, NY, USA, 2020, http://dx.doi.org/10.1145/3386164.3386176.

[23] F. Fabra, C. T. Calafate, J.-C. Cano, P. Manzoni, MBCAP: Mission based collision avoidance protocol for UAVs, in: 2018 IEEE 32nd International Conference on Advanced Information Networking and Applications, AINA, 2018, pp. 579–586, http://dx.doi.org/10.1109/AINA.2018.00090.

[24] F. Fabra, W. Zamora, J. Sangüesa, C.T. Calafate, J.-C. Cano, P. Manzoni, A distributed approach for collision avoidance between multirotor uavs following planned missions, Sensors 19 (10) (2019) http://dx.doi.org/10.3390/s19102404, URL https://www.mdpi.com/1424-8220/19/10/2404.

[25] A. Azzabi, K. Nouri, Path planning for autonomous mobile robot using the potential field method, in: 2017 International Conference on Advanced Systems and Electric Technologies, 2017, pp. 389–394, http://dx.doi.org/10.1109/ASET.2017.7983725.

[26] C. Huang, W. Li, C. Xiao, B. Liang, S. Han, Potential field method for persistent surveillance of multiple unmanned aerial vehicle sensors, Int. J. Distrib. Sens. Netw. 14 (1) (2018) 1550147718755069, http://dx.doi.org/10.1177/1550147718755069.

[27] D. Choi, K. Lee, D. Kim, Enhanced potential field-based collision avoidance for unmanned aerial vehicles in a dynamic environment, 2020, http://dx.doi.org/10.2514/6.2020-0487.

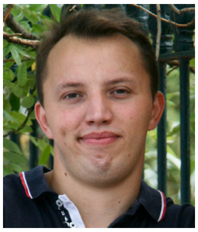
[28] C. Kownacki, L. Ambroziak, A new multidimensional repulsive potential field to avoid obstacles by nonholonomic UAVs in dynamic environments, Sensors 21 (22) (2021) URL https://www.mdpi.com/1424-8220/21/22/7495.

[29] L. Meier, QGroundControl, MAVLink micro air vehicle communication protocol, 2007, https://mavlink.io/en/. (Accessed 11 May 2019).

[30] OpenSim Ltd., OMNeT++, discrete event simulator, 2022, https://omnetpp.org/. (Accessed 7 December 2022).

[31] Navstar, Global positioning system standard positioning service performance standard, 2020, https://www.gps.gov/technical/ps/2020-SPS-performance-standard.pdf.

[32] G. Geraci, A. Garcia-Rodriguez, L. Galati Giordano, D. López-Pérez, E. Björnson, Understanding UAV cellular communications: From existing networks to massive MIMO, IEEE Access 6 (2018) 67853–67865, http://dx.doi.org/10.1109/ACCESS.2018.2876700.



Jamie Wubben is a researcher in the Networking Research Group (GRC) at the Universitat Politècnica de València (UPV), Spain. He holds a Master of Electronics and ICT Engineering Technology from the University of Antwerp (2019) and a Master in computer and network engineering from the UPV. Currently, He is doing his Ph.D. at the UPV; His research is focused on simulation and automatic control of UAV swarms.

Carlos T. Calafate is a full professor in the Department of Computer Engineering at the Technical University of Valencia (UPV) in Spain. He graduated with honors in Electrical and Computer Engineering at the University of Oporto (Portugal) in 2001. He received his Cum Laude Ph.D. degree in Informatics from the Technical University of Valencia in 2006, where he has worked since 2002. His research interests include ad-hoc and vehicular networks, UAVs, Smart Cities & IoT, QoS, network protocols, video streaming, and network security. He is ranked among the World's Top 2% Scientists, and also among the top 100 Spanish researchers in the Computer Science & Electronics field.

Juan-Carlos Cano is a full professor in the Department of Computer Engineering at the "Universitat Politècnica de València (UPV)" in Spain. He earned an M.Sc. and a Ph.D. in Computer Science from the UPV in 1994 and 2002, respectively. His current research interests include Wireless Communications, Vehicular Networks, Mobile Ad Hoc Networks, and Pervasive Computing.

Pietro Manzoni received the master degree in Computer Science from the "Università degli Studi" of Milan, Italy, in 1989, and the Ph.D. degree in Computer Science from the "Politecnico di Milano", Italy, in 1995. His research activity is related to the use of Mobile Wireless Networks to the design of dynamic systems. He is currently working on solutions for the Internet of Things focussing on LPWAN-based networks, and Pub/Sub systems. The overall focus is on the design of sustainable solutions, that is, solutions that try to consider at least one of sustainability's three main pillars: the economy, society, and the environment. He is currently a full professor of computer engineering at the "Universitat Politècnica de València", Spain. He is the coordinator of the Computer Networks Research Group (GRC) and a senior member of the IEEE.