**Priority-Aware Conflict Resolution for U-Space**

**Jesús Jover**, Aurelio Bermúdez * and Rafael Casado

Albacete Research Institute of Informatics, University of Castilla-La Mancha (UCLM), Campus Universitario s/n, 02071 Albacete, Spain; jesus.jover@uclm.es (J.J.); rafael.casado@uclm.es (R.C.)

* Correspondence: aurelio.bermudez@uclm.es; Tel.: +34-967-599-200

**Abstract:** In the context of the future urban mobility airspaces, U-space lays the foundation for unmanned aircrafts to be integrated with conventionally manned traffic to offer a multitude of services to citizens and businesses, such as urban public transport, goods movement, parcel delivery, etc. Given these extensive possibilities, it is estimated that thousands of aircrafts (or even more) from different owners will share common airspace volumes, and the probability that all of them will travel without colliding is extremely low if no means are put in place to prevent it. This paper aims to introduce the reader to the conflict management policy proposed in U-space and proposes a new version of the PCAN (Prediction-based Conflict-free Adaptive Navigation) algorithm to adapt to this policy. In particular, the new implementation of PCAN takes into account a more than likely classification of flights according to their priority. The evaluation of this priority-aware algorithm shows that it is still possible to avoid any conflict between the set of aircraft in flight, with an insignificant detriment to higher priority flights, while maintaining minimal detriment on average.

**Keywords:** Urban Air Mobility (UAM); Unmanned Aircraft System (UAS); UAS Traffic Management (UTM); U-Space; conflict detection and resolution

---

**1. Introduction**

Recently, the need for a regulation for an airspace open to UAV/UAS (Unmanned Aircraft Vehicles/Unmanned Aircraft Systems) has increased due to the development of different aircraft for a multitude of services and the concept of UAM/AAM (Urban Air Mobility/Advanced Air Mobility) [1]. Different organizations around the world began some years ago to establish the bases and the necessary knowledge to create a regulation to allow the incorporation of UAS and their applications within the civil airspace in a scalable, flexible, adaptable, and most importantly, safe way. In the US space we find NextGen UTM (UAS Traffic Management) [2], developed by the FAA, while in the European space we have U-Space [3,4], developed by Eurocontrol and the European Union (EU) under the umbrella of the SESAR Joint Undertaking partnership.

The U-Space ConOps (Concept of Operations) details the basic principles on the integration of unmanned aircraft in civil airspace, the requirements and need for their access, as well as the services that U-Space providers must offer to pilots and, ultimately, to any stakeholder wishing to fly any non-ATM (Air Traffic Management) aircraft. This ConOps does not represent a regulatory proposal, nor does it detail how U-space will end up working, as it is still under development.

The SESAR Joint Undertaking has just launched the AURA project [5], which aims at integrating the new U-space services to come with the current ATM systems. The objective is to generate a ConOps and validate the information exchange methods between UTM and ATM. In this way, airspace segregation is avoided, and interoperability is increased. In total there are 29 project participants, including drone operators, U-space service providers, ATM providers and authorities from different countries.

Together with AURA comes the EU 2021/665 regulation [6], which modifies the EU 2017/373 regulation [7], and whose main changes are:
Electronics 2022, 11, 1225

- A U-space airspace may be designated within manned airspace so that manned aircraft operate alongside unmanned ones;
- Air navigation service providers (ANSPs) are responsible for carrying out dynamic reconfiguration of U-space airspace to ensure that manned and unmanned traffic remain properly delineated;
- The necessary procedures and communication facilities must be in place to provide both manned and unmanned traffic with information on airspace reconfiguration.

Both AURA and the new regulation highlight the current need to develop, simulate and validate all the systems and services that must be present in U-space, such as tactical management, and which will start to become available in the short to medium term as UAS are integrated into different airspaces.

Among the services offered by U-space service providers, one of special interest in the community is that of conflict management. Here, a “conflict" is understood as a situation in which two or more UAVs are at a distance less than a minimum separation predetermined by regulation. This is a classic problem in traditional ATM [8], where it is called Conflict Detection and Resolution (CD & R), and there are already countless resolution proposals [9]. In short, the final purpose of conflict management is to avoid at all costs the occurrence of a collision between two aircraft.

One possible way of classifying conflict management techniques is to differentiate between those that obtain a priori a set of conflict-free routes or flight plans for the UAVs and those that detect conflicts during flight and resolve them by modifying the flight plan of the UAVs involved. Along the same lines, both UTM and U-space distinguish between strategic (pre-flight) and tactical (in-flight) conflict management. Other classifications take into account whether conflict control is centralized or distributed in each UAV, or whether there is communication between UAVs sharing the airspace [10].

From the perspective of tactical conflict management, Ref. [11] presents a new proposal for the navigation of autonomous UAVs in UAM scenarios that manages to avoid any conflict between them while executing their corresponding flight plans. The proposal, called PCAN (Prediction-based Conflict-free Adaptive Navigation), is based on predicting conflicts based on the estimation of the future position of the UAVs. To prevent the occurrence of each predicted conflict, PCAN proceeds to adapt the velocity vector of the UAVs involved, considering the airspace situation. PCAN works in a centralized way, from the position, velocity, and destination information of all UAVs flying over the airspace and is valid for any type of UAVs capable of hovering, such as single-rotor UAVs and multi-rotor UAVs (like the popular quadcopters).

PCAN can be used as baseline for the development of more advanced tactical conflict resolution algorithms, since it offers a very good performance in both low and high-density scenarios, introducing at the same time a discrete overhead in flight time and distance travelled.

U-Space considers the possibility of assigning different priorities to the set of UAVs in flight, depending on the type of operation they are carrying out. Under this assumption, in this work we propose a new version of the PCAN algorithm, which takes into account the priority assigned to each UAV in the airspace. As in similar state-of-the-art proposals that handle priorities, those UAVs with a higher priority are given priority treatment by the new implementation. In particular, such UAVs will have their velocity vector modified to a lesser extent than those UAVs with lower priority.

The rest of this work is structured as follows. Section 2 firstly introduces both conflict management and the priority policy, as defined in the U-Space ConOps, and then presents different conflict management approaches, including those that consider UAV priorities. Next, in Section 3 we briefly introduce the PCAN algorithm for tactical conflict management, detail the new priority-aware version of PCAN, and discuss some issues related to real-time implementations. After that, Section 4 compares the results offered by this new version with respect to the original one, as well as other well-known conflict management algorithms. Finally, in Section 5, we draw some conclusions and detail future works to be done.
2. Background

2.1. Conflict Management and Priorities in U-Space

U-space differentiates three conflict management levels, depending on when the conflict occurs:

- Strategic conflict management: consists of managing conflicts before UAS take-off. It is performed taking into account the specified flight plan. The routes to be followed by the UAS will be analyzed for potential conflicts and modifications will be proposed to resolve them;
- Tactical conflict management: consists of managing conflicts once the UAS are in flight. This is done by monitoring the airspace situation to avoid that aircraft fail to respect a minimum separation between them. This management can be carried out with the help of hardware and instrumentation, taking as the main input source the data reported by the UAS to the service provider. Through tracking and monitoring, the systems will be able to detect conflict situations and propose trajectory changes to the UAS through an emergency management service;
- Collision avoidance: if none of the previous levels have been able to avoid the conflict or if an unforeseen event has occurred that prevents the navigation plan from being followed, the UAS may be equipped with different systems to detect and avoid a mid-air collision (MAC) at the last moment.

Not all the UAS will have three-level conflict management. Depending on the airspace volume through which the aircraft wishes to transit, different levels will be required. In particular, U-Space defines a Very Low Level (VLL) airspace for UAV/UAS flight, which is decomposed into three volumes, X, Y, and Z. Strategic conflict management is only required for UAS that wish to fly in Y and Z volumes. Tactical conflict management is only required on Zu volumes. In Za volumes, tactical conflict management is carried out by the traditional Air Traffic Service (ATS).

On the other hand, when resolving conflicts, it is not always possible to deal with them in the fairest way (for example, by distributing the necessary deviation to avoid the conflict between the two involved UAS), and in some cases it should not be that way either. One of the key principles in the development of U-space is a fair and equal access for all users and companies wishing to operate, so that regardless of the person behind an aircraft, it should be treated with the same rights as any other, without distinction.

However, not all flights carried out in airspace should be given equal priority. Certain flights, given their need for urgency, must be prioritized over other services that do not require such treatment. This is the case of flights related to emergency services, such as air ambulances or search and rescue works, which should have greater freedom of movement and a more beneficial response when resolving conflicts. In this sense, U-space proposes under the name of “prioritization” different levels of priority, assigning a given priority to each of the services carried out by an aircraft. In the U-Space ConOps it is proposed to define 8 priority classes, numbered from 1 to 8, with a higher number being a lower priority (see Table 1). This classification is still a proposal and has not yet been validated, but the final version is expected to keep this idea.

Table 1. Prioritization proposed in U-space, depending on the type of operation.

<table>
<thead>
<tr>
<th>Priority</th>
<th>Type of Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1 (max)</td>
<td>Emergency—Air ambulance, Search and rescue, catastrophe</td>
</tr>
<tr>
<td>Class 2</td>
<td>Hospital (Hospital transport)</td>
</tr>
<tr>
<td>Class 3</td>
<td>Authorities (Police, other), Military</td>
</tr>
<tr>
<td>Class 4</td>
<td>Urgent Transport, Passenger transport</td>
</tr>
<tr>
<td>Class 5</td>
<td>Agriculture, Inspection, Infrastructure, Science, Weather</td>
</tr>
<tr>
<td>Class 6</td>
<td>Transport</td>
</tr>
<tr>
<td>Class 7</td>
<td>Video, Filming, TV, Empty flight (ferry flight)</td>
</tr>
<tr>
<td>Class 8 (min)</td>
<td>Leisure, Sports</td>
</tr>
</tbody>
</table>
In addition, within the same priority there may be thousands or millions of UAS, for example, in the transport class. In this case, a second prioritization consideration within the same class could be the relative agility of the UAS, as well as the number of maneuvers it must perform to resolve the conflict. In this sense, U-space also proposes a second prioritization system based on the number of maneuvers to be performed by a UAS. In this way, the UAS with the lowest maneuvering capacity will be the “highest priority” in terms of conflict resolution.

The existence of different priorities between flights should influence both strategic and tactical conflict management. These services offered by U-space service providers must take into account the priority of flights and their outcomes must be commensurate. For example, if two UAS, one with classified emergency operation and the other with classified transport operation, find themselves in a conflict situation, approaching a risky situation, it will be the latter who will have to reduce or modify its trajectory to avoid the situation, keeping the trajectory and speed of the more priority one unchanged.

In terms of tactical conflict management, another concept to consider is the reasonable time to act. In the U-Space ConOps, RTTA (Reasonable Time To Act) is defined as the window of time required prior to takeoff for the U-space services to perform the necessary service and flight plan management calculations. That is, this time window will be the one used by the services, for example, to check if different flight plans of different UAS cause conflicts. This RTTA should always be as close to take-off as possible, so that at the time of calculation the “most accurate and up-to-date picture of the future state of the airspace” is available. Thus, whenever flight plans are modified prior to RTTA, it must be ensured that there is no prioritization based on the order in which the flight operations were planned. Conversely, if a flight plan is added or modified in the RTTA time window, it will be this flight plan that must be adapted to the existing plans, protecting those that were submitted earlier, and therefore, being disadvantaged with respect to other plans.

2.2. Tactical Conflict Management and Priority-Aware Previous Proposals

Within the wide variety of tactical conflict management techniques proposed in the literature, there are strategies based on the calculation of the space of valid velocities that prevent any potential conflict between UAVs, such as the popular ORCA (Optimal Reciprocal Collision Avoidance) algorithm [12] and the BBCA (Bounding Box Collision Avoidance) algorithm [13]. Still, there are also numerous force-based strategies, such as APF (Artificial Potential Field) [14], or swarm intelligence algorithms, such as PSO (Particle Swarm Optimization) [15]. It is also worth mentioning the existence of a group of methods, referred to as “sense-and-avoid”, in which the UAV is equipped with special hardware (LiDAR type) that allows a fast response in case of imminent collision [16,17].

Focusing on velocity-based conflict avoidance methods, they are based on choosing the optimal velocity for a mobile agent (in our case, an UAV) from among all those velocities that avoid conflict with other mobile agents (called “valid velocities”). Each agent predicts its future position and that of nearby agents based on current velocities and, under this assumption, determines its new velocity according to some optimization criterion. In ORCA each agent computes a Velocity Obstacle (VO) with respect to each of the agents flying in its environment. The VO includes all the prohibited velocities for the UAV. ORCA guarantees collision-free navigation in sparse scenarios. In dense scenarios it is possible not to find a valid velocity. Linear programming techniques are used to solve this problem, with a medium to high computational cost [18].

BBCA exhibits a lower complexity than ORCA and therefore is more appropriate for real-time conflict management. Basically, in BBCA the set of valid velocities for each UAV is represented by a rectangular area. This area is modified in each execution of the algorithm (by very simple operations) to avoid conflict with each of the nearby UAVs. BBCA can avoid any conflict in scenarios with two UAVs, and most of them in denser scenarios, with a reasonably acceptable overhead on the distance traveled and the final flight time of the UAVs.
The mentioned PCAN algorithm presents a good tradeoff between computational cost and overhead in distance and time. To do this, a “optimal” velocity is assigned to each UAV. This velocity is known as direct velocity and follows the shortest path, at a maximum velocity, between the UAV current position and the next waypoint in its route. In each execution, PCAN predicts the future position of the set of UAVs in the airspace and decides which ones should modify their velocity vector, and which ones should not. For the former, PCAN generates new velocity vector candidates by modifying the direction of the direct velocity and by modifying its modulus but maintaining its direction and sense. The aim of this process is to alter the direct velocity as little as possible. Next section gives more details about the behavior of PCAN.

In the present work, and according to the U-Space ConOps “prioritization” recommendations [3], we assume that each UAV in the airspace has a priority level depending on the type of operation it is performing. Several conflict (or collision) avoidance methods have been proposed in the fields of robotics and UAVs that consider the use of priorities.

Prioritized planning [19] is a classic way of assigning trajectories to a set of robots. Each robot has a unique priority value, and the planning process proceeds sequentially from the highest-priority robot to the lowest one. For each robot a trajectory is planned that avoids both the static obstacles in the environment and the higher priority robots moving along the trajectories planned in the previous iterations.

In [20], the CD & R problem for multiple UAVs is formalized as a nonlinear optimization problem that is solved by considering UAV priorities among other constraints. Indeed, results show that UAVs with the lowest priorities take larger detours to avoid the conflict. Authors do not detail the way to assign priorities to UAVs and ignore the problem of the computational complexity associated to the resolution of these kind of problems.

The MBCAP protocol [21] avoids collisions between a set of UAVs following a planned mission. To do this, there is a negotiation process between the UAVs involved, in which low-priority UAVs will wait (stopped in the air) the needed time to avoid the collision. Priority is defined according to a unique ID assigned to each UAV (the highest priority UAV is the one with the highest ID value).

The work in [22] formulates the conflict resolution problem as a game model and uses a priority ranking mechanism to solve it. This ranking is established by an environmental risk assessment, the flight delay, the remainder fuel endurance, and the proximity to the destination. One more time, aircraft with lower priorities are the ones who must make the most sacrifices to avoid conflict.

The conflict resolution algorithm proposed in [23] is based on model predictive control (MPC). In each execution step, the method predicts potential conflicts between UAVs and provides new trajectories without collisions for the UAVs with lower priorities. Priorities are assigned to the UAVs involved in the conflict according to some rules, that have to do with the time to reach the conflict position, the distance to their destinations, and their speeds. The MPC approach is also used in [24].

The method described in [25] handles conflicts at strategic, pre-tactical (or pre-flight), and tactical levels. UAV priorities are randomly assigned. In case of conflicts at the first, second, and third level, different techniques (rerouting, ground delays, and hovering, respectively) are applied to UAVs with less priority for deconfliction.

The strategy presented in [26] also handles conflicts before and during the flight. First, UAV trajectories are computed using the Fast Marching Square planning algorithm, which guarantees a safety distance between UAVs. Then, a basic priority-based speed control mechanism is used to resolve possible conflicts when the UAVs are in the air. Finally, authors in [27,28] focus on tactical conflict management for U-Space assuming 4D trajectory-based operations. In line with previous works, in case of a potential conflict only the set of waypoints defining the trajectory of the UAV with lower priority is varied.
3. Managing Priorities in PCAN
3.1. PCAN General Behavior

Before presenting the general behavior of the original version of PCAN, we provide some definitions and assumptions. Let \( x = [p \, \dot{p}] \) be the dynamic state of a UAV, where vectors \( p = [p_x \, p_y] \in \mathbb{P} \) and \( \dot{p} = [v_x \, v_y] \in \mathbb{V} \), represent its current position and velocity, respectively. Let \( \mathbb{A} = \{a_1, a_2, \ldots, a_n\} \) be the set of UAVs in the airspace at time \( t \). Each element \( a = [x \, R] \in \mathbb{A} \) represents the status vector of a UAV, including its dynamic state and a route composed of several waypoints to its destination. Let \( R = \{w_1, w_2, \ldots, w_m\} \) be an ordered sequence of waypoints \( w_i \in \mathbb{P} \). Without loss of generality, we assume that all UAVs behave homogeneously, initially flying at a predetermined maximum velocity \( v_{\text{max}} \) and they are considering a safety radius \( r \) (which delimits a “protected zone” around them). Figure 1 shows an example.

![Figure 1. Two UAVs in the airspace.](image)

PCAN was designed with the goal of avoiding conflicts at all costs. For this, each execution of the algorithm predicts the occurrence of conflicts between the set of UAVs in the airspace by estimating their future positions.

The first task at the beginning of each execution of PCAN consists of assigning to each UAV in \( \mathbb{A} \) the direct velocity vector from its current position to the next waypoint on its route.

Then, PCAN goes through the UAVs in \( \mathbb{A} \) and, for each of them, it compares its velocity vector with that of the set of UAVs located at a shorter distance than the distance the UAV will cover before the next execution of the algorithm. Such a comparison allows determining the existence of future conflicts between each pair of UAVs. As an example, Figure 2 shows two UAVs whose trajectories lead to a conflict in the future.

Each time a future conflict is detected, the UAV under study can no longer use the direct velocity vector. An iterative process comes then into play to find a new velocity vector that avoids the conflict.
We define the priority of each UAV in the airspace by means of a value \( p \in \mathbb{N} \), so that \( 0 < p \leq p_{\text{low}} \). For convenience, lower values of \( p \) imply higher priorities (as in U-Space), being \( p_{\text{low}} \) the lowest possible priority. From now on, each UAV in the airspace is denoted by \( a = [x R p] \in A \).

To illustrate the desired behavior for the new version of PCAN, Figure 3 shows a scenario in which a conflict is predicted between two UAVs \( a_1 = [x_1 R_1 p_1] \), \( a_2 = [x_2 R_2 p_2] \in A \), with \( p_1 < p_2 \). Shaded circles indicate the moment when the protected areas of both UAVs start to overlap. In this situation, the algorithm should maintain the velocity vector for \( a_1 \) (that is, \( v_1 \)), and provide an alternative vector \( v_2' \) for \( a_2 \). In other words, the movement of the highest-priority UAV should not be affected.

The original implementation of PCAN considers the set of UAVs in the airspace in a certain order, and this order is very important, since those UAVs that are studied first will have to resolve a greater number of conflicts than those that are dealt with later (the former solve the conflict situations of the latter). Therefore, the earlier a UAV is treated, the more modifications in its velocity vector it will undergo. Thus, if we want that PCAN
behaves as in the example shown in Figure 3, it should process the set of UAVs according
to their priorities, from the lowest to the highest (in our case, from the highest $p$ value to
the lowest).

To assign different priorities to each UAV in the airspace we could use, for example, a
multi-level priority scheme, in which the first level represents the UAV class, as established
in the U-space ConOps (see Table 1). This means that all UAVs will have a value $p_{L1} \in [1, 8]$, where a lower value indicates a higher priority. Two UAVs with the same $p_{L1}$ are of equal
priority at this level. According to the ConOps, a second priority level would allow a priority setting to be established based on a second criterion beyond the class of the UAV, for example, the level of maneuverability of the aircraft. This would allow different priorities
to be set between UAVs of the same class. A third priority level like the second one could
be also added. Therefore, the second and third levels would have two values associated
$p_{L2}, p_{L3} \in [1, 100]$ (for example), where higher numbers indicate higher priorities.

Prior to UAV takeoff, initial priority values must be set at the levels considered. During
the flight, priority values may change due to U-Space service provider policies, and the new
values will be used in subsequent executions of PCAN. Consequently, if a UAV increases in
priority during flight (its $p$ value is reduced), from that moment it will be given priority
treatment. However, the overhead introduced globally in its path will be higher than if
this higher priority level had been assigned from the beginning. The same is true for a
priority reduction.

From the above considerations, we can detail the operation of the proposed PCAN
algorithm, based on the candidate implementation shown in Table 2.

<table>
<thead>
<tr>
<th>Table 2. Priority-aware version of the PCAN algorithm.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A = \text{PCAN}(A)$</td>
</tr>
<tr>
<td>01 assume $r, p_{low}$</td>
</tr>
<tr>
<td>02 for all $a \in (A)$ do</td>
</tr>
<tr>
<td>03 $[\mathbf{p} \sim R \mathbf{p}] = a$</td>
</tr>
<tr>
<td>04 $\mathbf{v} = \text{DirectNav}(a)$</td>
</tr>
<tr>
<td>05 $a = [\mathbf{p} \mathbf{v} R \mathbf{p}]$</td>
</tr>
<tr>
<td>06 end do</td>
</tr>
<tr>
<td>07 $p_{\text{level}} = p_{\text{low}}$</td>
</tr>
<tr>
<td>08 while $p_{\text{level}} &gt; 0$</td>
</tr>
<tr>
<td>09 for all $a \in A \mid p = p_{\text{low}}$ do</td>
</tr>
<tr>
<td>10 $[\mathbf{p} \mathbf{v} R \mathbf{p}] = a$</td>
</tr>
<tr>
<td>11 if AirspaceConflicts($\mathbf{v}, a, A$)</td>
</tr>
<tr>
<td>12 $\mathbf{v} = \text{ValidVelocity}(a, A)$</td>
</tr>
<tr>
<td>13 $a = [\mathbf{p} \mathbf{v} R \mathbf{p}]$</td>
</tr>
<tr>
<td>14 if $\mathbf{v} = 0$</td>
</tr>
<tr>
<td>15 $p_{\text{level}} = p_{\text{low}} + 1$</td>
</tr>
<tr>
<td>16 break</td>
</tr>
<tr>
<td>17 end if</td>
</tr>
<tr>
<td>18 end if</td>
</tr>
<tr>
<td>19 end for</td>
</tr>
<tr>
<td>20 $p_{\text{level}} = p_{\text{level}} - 1$</td>
</tr>
<tr>
<td>21 end while</td>
</tr>
</tbody>
</table>

As in the original version of PCAN, the first task to perform consists of assigning a
direct velocity to the destination to all the UAVs in the airspace (lines 02–06 in Table 2).
Next, PCAN must process one at time all the elements in $A$. As indicated above, this
processing must be done by priority levels, starting with the UAVs with the lowest priority
($p = p_{\text{low}}$) (line 07).

In the core part of the algorithm, from the current positions and velocity vectors
of each pair of UAVs, a system of equations is built whose resolution provides the time
instant (if any) at which the separation between both UAVs will be less than the safety
radius ($r$). This check is performed by using the AirspaceConflicts() function (line 11), whose implementation is detailed in [11].

In case of conflict, PCAN will use the ValidVelocity() function (line 12) to determine a new velocity vector for one of the UAVs involved, so that this new velocity does not generate any conflict, not only with the other UAV in the pair, but with any other UAV in the airspace. The set of velocity vectors provided after processing all the possible pairs of UAVs guarantees that, at least before the next execution of the algorithm, no UAV will produce a conflict with any other given UAV. In other words, a conflict-free scenario.

In addition, PCAN always aims to optimize the distance traveled and the time in flight of the UAVs, since autonomy is one of the major limitations of these kind of aircraft. Therefore, to predict the existence of conflicts, PCAN does not actually use the velocity vector with which UAVs are currently moving but uses the direct velocity, that is, the one that covers in the shortest time the shortest path to the next waypoint in the UAV route. That is the purpose of the invocations to the DirectNav() function at the beginning of the algorithm (line 04). Since PCAN introduces modifications to the velocity vector, the fact that these modifications start from the direct velocity makes the solutions more efficient than if these modifications were made on a previously modified velocity vector.

The strategy used by the ValidVelocity() function to modify the UAV velocity vector once the existence of the conflict has been determined is an iterative process that is executed until a solution that solves any conflict is achieved or until a limiting number of iterations, called $m$, is reached. At each iteration, PCAN generates three new candidate vectors: (1) a vector modified in modulus (with a lower velocity); (2) a vector modified in clockwise direction; (3) a vector modified in counterclockwise direction. In the case of the first iteration, the three velocity vectors generated are computed from the direct velocity. The strategy consists in gradually introducing small variations in the vectors considered in the previous iteration, so that, in case one vector resolves all conflicts, it is as close as possible to the direct velocity.

The parameters indicating the modification introduced in each iteration are two: (1) $coef$, which is a coefficient by which the velocity modulus of the previous iteration is multiplied; (2) $\alpha$, which indicates the angular displacement to be applied with respect to the velocity of the previous iteration. Both parameters are detailed in [11].

As stated, the search procedure for a new velocity vector concludes if after $m$ iterations a valid (conflict-free) solution has not been found. This can happen in very crowded airspace regions. In this situation, the ValidVelocity() function provides a null velocity to the UAV, which in practice means that it will stop in the air until it can move forward again (in a later execution of the algorithm). Note that assigning a null velocity to a UAV triggers an additional operation (affecting computational cost), consisting in considering again for analysis the subset of UAVs that have already been processed (lines 14–17). This is because the prediction they had made is no longer valid, since it was made by assuming that the UAV that now has been stopped would follow the direct velocity vector.

The PCAN algorithm is executed periodically, based on a parameter called $t_{nav}$. As mentioned, the solution offered by the algorithm ensures that no conflict will occur, at a minimum, until its next execution (in $t_{nav}$ seconds). An airspace bounding has been incorporated that allows, if desired, that for each UAV only those UAVs that might conflict with it before the next execution of the algorithm are considered, instead of taking into account all the UAVs in the airspace (regardless of their distance from the UAV under study). Obviously, this can save a lot of unnecessary computations and therefore speed up the algorithm execution time. This bounding is done based on the $BR$ (boundary radius) parameter, which allows to enlarge or reduce the size of the circular airspace region around the UAV under study considered to predict conflicts.

### 3.3. Real-Time Considerations

The PCAN algorithm is executed in a centralized way (for example, in a U-Space ground station). All the computations are based on a global and updated knowledge of the
airspace situation, that is, we assume that the entity executing PCAN knows the current state and route of each UAV. This information can be known either by the transmission of periodic position and velocity reports or with the support of a tracking service. Both options are available in the U-Space ConOps. After each execution, PCAN transmits velocity updates back to the UAVs. We should consider the impact of all these communications in a real-time deployment.

In Figure 4, two consecutive executions of PCAN ($n$ and $n + 1$) are represented. $t_c$ is the time it takes to execute the algorithm (which will be estimated in Section 4.3). There are two UAVs $a_1, a_2$ with different reporting periods ($t_{rep1}$ and $t_{rep2}$). Red and green vertical segments indicate the sending of report messages $M_i = [p_i, v_i]$, and dark blue vertical segments indicate the sending of new velocities to the UAVs ($M_{PCAN}$). We assume that all the transmissions ($M_i$ and $M_{PCAN}$) take a time $t_{rep}$.

![Figure 4. PCAN time behavior.](image)

It is quite possible that the sending of position reports is not synchronized with the execution of the algorithm. Therefore, before each run, PCAN must “correct” the information of all UAVs, by considering the time elapsed from the moment the reports were sent ($\Delta t^n_i$ for execution $n$) and the reported positions and velocities ($p_i$ and $v_i$).

For the calculation of $\Delta t^n_i$ we assume that report messages include a timestamp that informs about the exact time at which the report has been generated. We also assume that UAVs and U-Space share a common time, provided, for example by accurate GPS receivers.

In this way, the corrected position of UAV $a_i$ for execution $n$ of PCAN may be expressed as $p_i = p_i + v_i(\Delta t^n_i + t_c + 2t_{rep})$. This is the position value that the algorithm will employ to guarantee a conflict-free situation at least for the next $t_{nav}$ seconds.

4. Performance Evaluation

In this section we provide a comparative evaluation of the priority-aware version of PCAN whose behavior has just been described. All these results provided below have been obtained by simulation. The simulation model has been developed in Matlab 2021b, by making use of the resources for object-oriented programming offered by this platform.

To obtain the results shown below, we have considered in all cases a $5 \times 5$ km airspace region, with different density of UAVs in flight. Figure 5 shows and example. In each simulation, UAV routes are randomly chosen, considering that between the initial position and the final position of each route there must be a distance greater than 1 km, and that both positions must be at least 100 m away from the edges of the flight region.

As simulation parameters, in all cases we have set $v_{max} = 50$ km/h, $r = 40$ m, $t_{nav} = 3$ s, and $\tau = 0.3$ s. The $\tau$ parameter is related to the modeling of the dynamic behavior of the UAV, and it is detailed in [11]. Precisely, such a dynamic behavior causes that the safety radius ($r$) has been increased by 8.34 m. Regarding the specific parameters of the two versions of PCAN (with and without priorities), in all cases we have set $BR = 3v_{max}t_{nav} + 2r$, $m = 50$, $coef = 0.95$, and $a = \frac{\tau}{m}$ rad.
Figure 5. Example of simulated random scenario showing the route followed by 100 UAVs (upper view). Numbers represent initial positions and circle represent current positions and safety areas. Units on axes are meters.

The simulation tool generates the priorities for each UAV also randomly. For the simulations presented here, only the first level of priority has been set (the $p_1$ value), so for the second and third levels, all UAVs have the same value. The priorities follow the distribution shown in Table 3. The probability for each of the classes has been chosen by estimating the number of UAVs that could be present in the future in each of them.

<table>
<thead>
<tr>
<th>Priority Class</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1</td>
<td>2%</td>
</tr>
<tr>
<td>Class 2</td>
<td>5%</td>
</tr>
<tr>
<td>Class 3</td>
<td>3%</td>
</tr>
<tr>
<td>Class 4</td>
<td>10%</td>
</tr>
<tr>
<td>Class 5</td>
<td>15%</td>
</tr>
<tr>
<td>Class 6</td>
<td>50%</td>
</tr>
<tr>
<td>Class 7</td>
<td>10%</td>
</tr>
<tr>
<td>Class 8</td>
<td>5%</td>
</tr>
</tbody>
</table>

4.1. PCAN Versus PCAN with Priorities

This first set of results aims to check the effect of managing UAV priorities in PCAN. For each airspace configuration we have compared the two versions of the PCAN algorithm described in this work, that is, the original implementation (“PCAN”) and the implementation that considers UAV priorities (“PCANprior”), to the direct algorithm, that is, the one in which UAVs always use the direct velocity to reach the next waypoint in the route. The direct algorithm (or direct method) does not manage eventual conflicts, but it represents the shortest and fastest path to destination.

For the simulations presented in this subsection, we have considered random scenarios with 100 UAVs in flight. In particular, a total of 100 scenarios have been generated, each of which has been simulated for each algorithm. In total, 300 simulations were executed.

First, we have analyzed the impact of working with priorities on the distances covered by the UAVs. Figure 6 shows in its ordinate axis the average increment in distance with
respect to the distance covered when the direct algorithm is in place. We can observe that, as expected, all the UAVs suffer a similar distance increment (about 4.5%) when the algorithm does not consider UAV priorities. However, with priorities, we see that higher priority classes suffer a much smaller increase in distance than lower priority classes. Specifically, the highest priority UAVs experiment an increase in distance traveled of less than 0.5%, while the lowest priority UAVs see an increase in distance traveled of up to 8%.

![Figure 6](image-url)  
**Figure 6.** Relative increment in distance, compared with the direct algorithm, by UAV priority classes.

We also have analyzed the increase in the time spent by UAVs in flight. Figure 7 shows the results. We observe results analogous to the previous ones. Without priority management, all the UAVs obtain a similar increase in flight time (around 8%). On the contrary, with priorities, we see that the higher the priority, the lower the increase in flight time. Specifically, UAVs with the highest priority experiment an increase in time in the air of less than 0.4%, while those with the lowest priority see an increase of up to 17%.

![Figure 7](image-url)  
**Figure 7.** Relative increment in time, compared with the direct algorithm, by UAV priority classes.

### 4.2. **PCAN versus Other Techniques**

Next, we compare the behavior of the two versions of PCAN described in this work to the direct algorithm and to other conflict management techniques of the state-of-the-art (specifically, APF and BBCA), by considering again multi-UAV scenarios, but with different densities. In particular, we have considered 10 UAV densities, by varying the number of UAV in flight from 10 to 100. For each of these 10 densities, 24 different random scenarios have been generated, and each scenario has been simulated for the five mentioned algorithms, resulting in a total of 1200 runs of the simulation model.

The abscissa axis in the plot of Figure 8a indicates the number of UAVs in the flight region. This will be the same in following plots. In contrast, the ordinate axis shows the mean number of conflicts produced and the corresponding standard deviation. For the
direct method, conflicts increase exponentially with the number of UAVs in the region. APF manages to resolve isolated conflicts, but its performance decreases as the UAV density increases. BBCA significantly reduces the number of conflicts but does not eliminate them (although it cannot be appreciated in this plot). Finally, both PCAN versions successfully resolve all the conflicts.

Figure 8. Number of conflicts: (a) not solved conflicts (mean and deviation); (b) solved conflicts (%).

Figure 8b shows the same results, but now from the point of view of the percentage of solved conflicts. We can clearly see that both versions of PCAN and BBCA outperform APF. The direct method is not plotted since it does not avoid any conflict. As in the previous subsection, this method will be used as a reference in the plots below.

Figure 9a shows the distance traveled by the UAVs, in absolute terms, as a function of the UAV density. As expected, when applying the direct method, the distance traveled by each UAV does not vary with its number, while the standard deviation decreases. BBCA introduces large detours in UAV routes, causing a linearly increase in the distance covered. On the contrary, PCAN and APF algorithms increase the distance traveled by the UAVs, resolving the conflict in a more efficient way.

Figure 9. Distance traveled: (a) absolute values (mean and standard deviation); (b) relative increase with respect to the direct algorithm.

In Figure 9b we can see the relative increase in the distance traveled by the UAVs with respect to the direct algorithm. With BBCA, increment goes from 5% to 38%. PCAN algorithms follow the same behavior; they increase the distance traveled from 0% to about 5%, causing a minor increment. Finally, APF behaves very similar to PCAN, increasing a bit less the distance, but at the cost of not being conflict-free.

Figure 10a shows the flight time of UAVs, in absolute terms, as a function of the UAV density. For the direct method, flight time does not depend on the number of UAV. Again,
BBCA introduces a large increment in time. APF also entails increments, but to a lesser extent. PCAN algorithms introduce the lowest increments.

Figure 10. Flight time: (a) absolute values (mean and standard deviation); (b) relative increase with respect to the direct algorithm.

Finally, Figure 10b compares the flight time for the conflict management techniques considered with respect to the direct algorithm. In the case of the PCAN algorithms, the increment is only about 5% in the densest scenarios.

4.3. Computation Time

To conclude the analysis, we studied the time required by the direct, BBCA, APF and PCAN (with and without priorities) algorithms. Figure 11 shows the time consumed to produce an output for every UAV each time the navigation computation is executed. Results were generated with an Intel i9-10900KF@3.7GHz, using one core.

Figure 11. Average time required by the navigation algorithms.

As shown, considering a priority system in PCAN does not affect the computational cost of the algorithm. For low-density scenarios, PCAN works very well, exhibiting better behavior than BBCA. As the density increases, PCAN requires more time to solve all the potential conflicts produced. Even so, the computation time needed in PCAN does not represent a big bottleneck. In scenarios with 100 UAVs, the time employed is less than 0.8 ms. Note that these are extremely dense scenarios, where conflict avoidance is a complex task.

5. Conclusions and Future Works

The future European airspace for unmanned aircraft (U-Space) incorporates in its concept of operation the possibility of classifying UAV flights according to their priority. This paper describes a new implementation of the PCAN algorithm, previously proposed for conflict avoidance in this scenario, which assumes the existence of these priorities. In
each execution, PCAN detects future conflicts between UAV pairs by using current velocity vectors and avoids these events by modifying the vector velocity of UAVs. This process is executed in a specific order. In the new version of PCAN, low-priority UAVs are processed first. As a result, high-priority UAVs are much less affected in their displacement since their velocity vectors suffer less changes. The performance analysis carried out has proven this point.

The analysis also shows that managing priorities does not impact the good performance of the PCAN algorithm. The overhead in both distance covered and fly time is unaffected. Results are quite similar to the ones presented in previous studies: an increase in distance and time of about 5% with respect to the direct algorithm in the densest scenarios. This makes the PCAN proposal suitable for the deployment of future tactical conflict resolution services in U-Space, even when the scenario contemplates different UAV priorities.

As future work, we firstly plan to review the internal behavior of PCAN, in order to reduce the computational cost it exhibits in dense airspaces. We also want to tackle the conflict management problem from a strategical point of view.

**Author Contributions:** Conceptualization, A.B. and R.C.; methodology, J.J., A.B. and R.C.; software, J.J. and R.C.; validation, J.J., A.B. and R.C.; formal analysis, J.J.; investigation, J.J.; resources, J.J., A.B. and R.C.; data curation, J.J.; writing—original draft, J.J. and A.B.; writing—review and editing, J.J., A.B. and R.C.; visualization, J.J.; supervision, A.B. and R.C.; project administration, A.B. and R.C.; funding acquisition, A.B. and R.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Spanish Ministerio de Ciencia, Innovación y Universidades (MCIU) and European Union (EU) under RTI2018-098156-B-C52 grant, by the Junta de Comunidades de Castilla-La Mancha (JCCM) and EU through the European Regional Development Fund (ERDF-FEDER) under SBPLY/19/180501/000159 grant, and by the Universidad de Castilla-La Mancha under 2021-GRIN-31042 grant.

**Conflicts of Interest:** The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

**References**


13. Sánchez, P.; Casado, R.; Bermúdez, A. Real-time collision-free navigation of multiple UAVs based on bounding boxes. Electronics 2020, 9, 1632. [CrossRef]


17. Park, J.; Cho, N. Collision avoidance of hexacopter UAV based on lidar data in dynamic environment. Remote Sens. 2020, 12, 975. [CrossRef]


23. Chen, H.X.; Nan, Y.; Yang, Y. Real-time conflict resolution algorithm for multi-UAV based on model predict control. Algorithms 2019, 12, 47. [CrossRef]


