Fifteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2023)

The Curse of Time Horizon in Detect & Avoid Algorithms

David GIANAZZA, Richard ALLIGIER, and Nicolas DURAND
ENAC
University of Toulouse
Toulouse, France
{lastname}@recherche.enac.fr

Abstract—**This paper introduces a centralized collision avoidance algorithm based on a Differential Evolution, for the purpose of studying the time horizon effect, a pathological behavior previously identified in Detect & Avoid (D&A) decentralized algorithms based on geometric methods. This pathological behavior, called time horizon effect, is most likely to occur during constant-speed encounters, when the lateral maneuvers issued by the D&A system postpone the crossing of trajectories beyond the horizon of conflict detection by maneuvering the flights toward parallel tracks. In such cases, the flights might end up locked on parallel tracks, missing their destination.**

The proposed centralized algorithm selects the best direction changes at each time step for all flights, with the primary objective to maintain a minimum separation between the flights.

We show that the time horizon effect also occurs when using such a centralized optimization algorithm having full knowledge of the flight intents, and propose mitigating strategies. This suggests that the horizon effect is not related to the distributed nature of many D&A methods found in the literature, but is rather a more general effect due to the myopic nature of the decision process.

Keywords—**Unmanned Aerial Systems, detect & avoid, collision avoidance, time horizon effect, differential evolution**

NOTATIONS

τ	Time horizon (or anticipation) for the conflict detection
Δ	Standard lateral separation
ψ_i	Current direction of flight f_i, between 0 and 2π
α_i	Maneuver of flight f_i, expressed as the angular deviation from its current direction ψ_i. $\alpha_i \in [-\pi, \pi]$
ϕ_i	Preferred direction of flight f_i towards its destination. $\phi_i \in [0, 2\pi]$
β_i	Angle between the candidate direction $\psi_i + \alpha_i$ and the preferred direction ϕ_i. $\beta_i \in [-\pi, \pi]$.
$x = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{pmatrix}$	Vector of decision variables for a problem involving n flights.

I. INTRODUCTION

The traffic of Unmanned Aerial Systems (UAS) is expected to increase dramatically in the near future [1], posing a number of potential safety issues to the Air Traffic Management (ATM) system. In the current ATM system, trajectory conflicts are solved by human controllers that might not be able to cope with the number and variety of unmanned systems that will fly soon.

Several solutions are currently envisioned to address this issue. Among the most popular is the idea to delegate the separation task to the UAS themselves. In this distributed approach, each UAS would embark a Detect & Avoid (D&A) logic [2] that would ensure self-separation from the other UAS, and possibly from the commercial flights sharing the same airspace, if any. Actually, segregating the UAS traffic from the commercial one might be safer: The disparity of airspeeds between UAS and commercial flights and the lack of knowledge of the each other's intent can make it very difficult for the UAS to maintain a safe separation with the commercial flights [3].

In any case, there remains the problem of self-separation among UAS, and the problem of certifying – or at least providing convincing scientific evidence – that such a distributed system would actually be safe, robust and efficient.

Providing some guarantees on the Detect & Avoid (D&A) algorithms is central to this process. The primary objective of a D&A algorithm is to ensure collision avoidance. This objective is usually achieved by extrapolating the current positions of the own and surrounding flights, and by deciding evasive vertical or lateral maneuvers when a loss of separation is anticipated within a given finite time horizon. To be safe and efficient, such algorithms should take into account a number of factors related to the distributed nature of the decisions made by the UAS, the uncertainties on the positions and velocities of the other UAS, and the coordination of the maneuvers, either explicitly through the broadcasting of messages, or implicitly through a shared logic.

A very common belief concerning D&A is that a system that ensures collision avoidance among the flying agents is a safe one. For example, the safety requirements suggested in [4] focus solely on two major accident types related to airborne operations: mid-air collisions and ground impact. Safety cases for UAS (*e.g.* [5]) mostly focus on ensuring separation and preventing mid-air collisions. Consequently, one might think that implementing a D&A logic that is guaranteed to prevent mid-air collisions is safe enough for UAS operations. This might not be the case, however.

In the following, we focus on a pathological behavior, called time horizon effect, that was – to our knowledge – first identified by Durand in [6], [7] when applying distributed geometric anti-collision methods initially designed for robots to aircraft flying at constant speeds or having very limited acceleration and speed ranges. In some cases, the flights end up locked in parallel tracks, and miss their destination, which is a critical issue both in terms of safety and efficiency. It was initially suggested in [6] that the distributed nature of the D&A logic and the partial knowledge that each flight has on the others' intents might be the underlying cause of the effect, or at least that it might be an obstacle to solving this issue, and that a centralized approach would solve it.

In this paper, we propose a centralized anti-collision algorithm based on Differential Evolution. This optimization algorithm has a full view of all the traffic and computes optimal direction changes at each time step, with the primary objective to avoid trajectory conflicts (i.e. anticipated separation losses), and with the secondary objective to guide each flight toward its destination. These are exactly the same objectives as in the autonomous approach of [6] and other similar approaches, except that we optimize an objective function at each time step, instead of relying on a distributed geometric logic coordinating the maneuvers.

In this study, we show that the time horizon effect still occurs in such a centralized context, which suggests that simply adopting a centralized approach is not sufficient to prevent it. We also examine how the horizon effect is influenced by the design of the objective function, and by the fact that maneuvers are shared among conflicting flights or not. Be it in a centralized or distributed context, the main issue is that the D&A algorithm does not plan beyond the horizon of the conflict detection, and that its primary objective is to maintain separation, not to reach the destination. This suggests that additional strategies such as [8] might be necessary to help flights to cross paths. Such strategies should plan beyond the horizon of the collision avoidance logic.

The rest of the paper is organized as follows. Section II gives some background on the collision avoidance problem in air traffic management. Section III formalizes the collision avoidance problem as an optimization problem. The centralized algorithm used to solve this problem is described in Section IV. The time horizon effect is shown on a case study with two flights in Section V, where we also suggest some strategies to reduce its occurrence. Results on dense traffic scenarios are then provided in Section VI. Section VII concludes the paper.

II. Background on Anti-collision, Conflict Resolution, and Detect & Avoid

Several concepts coexist in Air Traffic Management (ATM) that deal with the objective of avoiding mid-air collisions, usually by maintaining a safe separation between the flying aircraft or UAS.

The Traffic Alert and Collision Avoidance System (TCAS), also known as ACAS (Airborne Collision Avoidance System) [9] is an airborne system deployed on commercial aircraft, independent from the Air Traffic Control, that alerts the pilots on the traffic at proximity and gives resolution advisories to avoid collisions. Traffic alerts and resolutions advisories are issued less than a minute ahead of the anticipated collision. In the current version, only vertical advisories are issued. Future versions (TCAS III, ACAS III, ACAS-X) are envisioned, that would advise lateral maneuvers as well as vertical ones. Research on next-generation ACAS explore new anti-collision algorithms, with a number of approaches including Dynamic Programming [10], deep reinforcement learning [11], [12], or deep neural networks [13].

Air traffic controllers detect and solve trajectory conflicts (*i.e.* anticipated losses of separation) several minutes ahead of the potential loss of separation, before the TCAS triggers. Many research have been conducted on the automation of the conflict resolution task. They include centralized global optimization approaches based on genetic algorithms [14], [15], mixed-integer linear programming [16]–[18], hybrid non-linear and mixed-integer programming [19], constraint programming [20], and hybrid methods based on the cooperation of combinatorial solvers [21]. Autonomous, distributed approaches have also been proposed, using sliding forces to coordinate the aircraft maneuvers [22], potential fields [23], electric repulsion forces [24] which inspired the Modified Voltage Potential (MVP) of the Airborne Separation Assurance System (ASAS) in [25]. ASAS was tested in the Mediterranean Free Flight experimentations [26].

With the rapid development of Unmanned Aerial Systems (UAS), and the research being conducted on new vehicles for Urban Air Mobility (UAM), the need for a decentralized separation assurance system becomes more apparent. Some of the approaches developed for the next-generation ACAS or in the context of Free Flight could be envisioned for UAS self-separation, as well as a variety of other approaches [27], including multi-agent reinforcement learning methods [28]–[31], a geometric implementation of the Right of Way rules [32], or other geometric methods initially developed for robot anti-collision.

Anti-collision has been a subject of study in the field of robotics for a long time. The Velocity Obstacles (VO) method introduced in [33] allows a mobile robot to avoid static obstacles, but it does not work correctly when multiple agents actively maneuver to avoid each other. To cope with this coordination problem, Van den Berg *et. al.* introduced the Reciprocal Velocity Obstacles (RVO) in [34], and then the Optimal Reciprocal Collision Avoidance (ORCA) in [35]. The principle of the ORCA method is to move the relative velocity vector of conflicting robots outside a reciprocal velocity obstacle materializing the domain where a collision would occur in a given time horizon. ORCA guarantees that the maneuvers are implicitly coordinated among all the robots.

These anti-collision methods, initially designed for robots, have inspired a number of methods for air traffic separation. Balasooryan in [36], or d'Engelbronner *et. al.* in [37] have proposed methods inspired from the Velocity Obstacle (VO) algorithm. Similarly to the VO method, these methods do not coordinate the maneuvers of the flying aircraft or UAS. Snape and Manocha [38] extend the ORCA 2D-model – which does coordinate maneuvers – to the 3-dimensional space. They consider spherical protection volumes around the aircraft, which does not account for the different vertical and horizontal standard separations currently employed. In [3], Alligier *et. al.* propose a more realistic 3D-model inspired from ORCA, but only in the context of a $1 - vs - n$ anti-collision problem, where one UAS tries to avoid uncooperative commercial flights.

In [6], [7], Durand first identified the pathological time horizon effect that is the focus of the current paper, and that may occur when ORCA is applied to aircraft or UAS flying at near-constant speeds: A trajectory conflict might be solved simply by postponing its resolution beyond the time horizon of the conflict detection. In such situations, some flights might be set on parallel courses, never reaching their destinations. To mitigate this problem, Durand proposed a Constant-Speed Optimal Reciprocal Collision Avoidance (CS-ORCA) algorithm for aircraft and UAS.

Alligier *et. al.* showed in [8] that CS-ORCA does not completely solve the parallel-track issue, and introduced a Dual-Horizon Reciprocal Collision Avoidance (DH-ORCA) that showed better performances.

In the current paper, we study the time horizon effect in the context of a centralized approach, where collision avoidance is considered as a global optimization problem to be solved by finding the optimal velocity directions, for all flights, at each time step.

III. Collision Avoidance as an Optimization Problem

As we have seen in the previous section, many anti-collision algorithms are implemented as distributed logics where decisions are taken separately onboard each aircraft/UAS. Here, we have chosen a centralized approach to emphasize the fact that the horizon effect – which is the main focus of our study – is not necessarily related to the distributed nature of the logic. It may occur even when an optimization algorithm with a global view of the traffic has full control of all the flights.

In order to exhibit this horizon effect, we consider specific scenarios where all aircraft or UAS fly at the same constant speed at a same altitude. The centralized anti-collision algorithm operates in the horizontal plane only, controlling the directions of the velocity vectors at every time step. In our scenarios, time is discretized with a time step δt (5 seconds in our experiments).

For a situation involving n flights at a given time t_c (current time step), the vector of decision variables is $x = (\alpha_1, \ldots, \alpha_n)^T$, where α_i is a lateral maneuver expressed as the angular deviation of flight f_i from its current direction ψ_i.

At each time step, we would like to find an optimal value for x, i.e. optimal direction changes, so that all flights can reach their destination as quickly as possible while remaining separated one from the others.

In order to prevent separation losses, the optimization algorithm will try to solve the trajectory conflicts detected within a finite time horizon τ, considering clusters of conflicts such as the one illustrated on Fig. 1. Conflicts are formally defined in definition III.1.

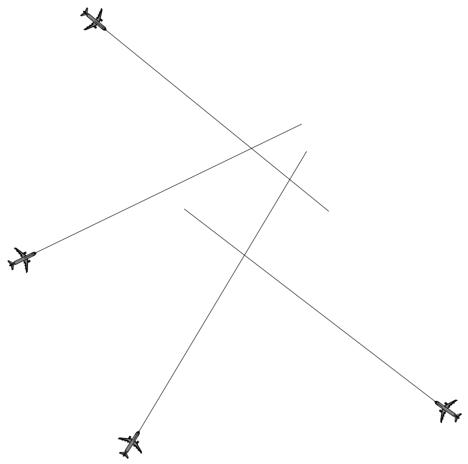
Figure 1: Conflict detection within a finite time horizon, assuming each flight follows a straight direction.

Définition III.1 (Conflict between two flights)**.** *Given a time horizon τ, a standard horizontal separation Δ, and considering two flights f_i and f_j at current time t_c, there is a conflict between f_i and f_j if and only if:*

$$\exists\, t \in [t_c, t_c + \tau] \quad \mathit{dist}(P_i(t), P_j(t)) \leq \Delta \tag{1}$$

where dist is the distance in the horizontal plane and $P_i(t)$ and $P_j(t)$ are the extrapolated positions of flights f_i and f_j at time t.

Conflicts occurring in a traffic situation can be characterized by their number of occurrences and their severity, both in terms of duration and worst infringement of the standard separation.

Définition III.2 (Conflict occurences)**.** *Denoting $\mathcal{C}$ the set of pairs (i, j) such that f_i and f_j are in conflict within the time horizon τ, the cost related to the conflict occurrences will simply be expressed as $N_c = |\mathcal{C}|$, i.e. the cardinal of $\mathcal{C}$.*

Définition III.3 (Conflict severity, in terms of lateral separation)**.** *Let us denote $d_{closest}(i, j)$ the closest lateral separation achieved by two flights f_i and f_j within the time horizon τ. If $d_{closest}(i, j) \leq \Delta$, where Δ is the standard separation, then the two flights are conflicting and the severity of the conflict is defined as follows:*

$$S_{dist}(i, j) = \frac{\Delta - d_{closest}(i, j)}{\Delta} \tag{2}$$

Définition III.4 (Conflict severity, in terms of duration)**.** *Considering a conflict occurring between two flights f_i and f_j within the time horizon τ, the severity of the conflict in terms of duration is defined as follows:*

$$S_{time}(i, j) = \frac{\mathit{conflict\_duration}(i, j)}{\tau} \tag{3}$$

where conflict_duration(i, j) is the duration of the conflict (bounded by τ, as potential future separation losses occurring beyond time $t_c + \tau$ are not accounted for).

The primary objective of the centralized optimization algorithm is to prevent separation losses by avoiding conflicts. If conflicts are unavoidable, the algorithm should favor solutions where conflicts are the less severe, in terms of lateral distance, and as short as possible. For this purpose, let us define two cost functions related to the conflict severity in terms of distance and time.

Définition III.5 (Cost function related to conflict severity in distance)**.** *Considering a situation involving n flights, and a vector $x = (\alpha_1, \dots, \alpha_n)^T$ of candidate lateral maneuvers, the overall conflict severity in terms of lateral separation infringements is defined as the worst separation infringement among all conflicting pairs of flights, assuming each flight f_i follows its candidate direction $\psi_i + \alpha_i$:*

$$C_{S_{dist}}(x) = \max_{(i,j)\in\mathcal{C}} S_{dist}(i,j) \tag{4}$$

where $\mathcal{C}$ is the set of conflicting pairs of flights.

Définition III.6 (Cost function related to the duration of conflicts)**.** *The overall conflict severity in terms of duration of the conflicts is defined as the average value of S_{time} over all conflicts:*

$$C_{S_{time}}(x) = \frac{1}{N_c} \sum_{(i,j)\in\mathcal{C}} S_{time}(i,j) \tag{5}$$

where $\mathcal{C}$ is the set of conflicting pairs $(i, j$ such that flights f_i and f_j are in conflict.

Also, each aircraft or UAS should fly toward its destination as directly as possible, while avoiding conflicts. With our model, this can be expressed very simply by the fact that the velocity of each flight f_i, once corrected by the assigned maneuver α_i, should preferably be pointing toward its destination. This can be expressed through the following angular deviation cost.

Définition III.7 (Angular deviation cost)**.** *Considering a situation involving n flights, and a vector $x = (\alpha_1, \dots, \alpha_n)^T$ of candidate lateral maneuvers, we define the following cost function related to the deviations from the preferred direction toward destination.*

$$C_{angle}(x) = \sum_{i=1}^{n} |\beta_i|^\eta \tag{6}$$

where η is a chosen parameter and β_i is the angle between the candidate direction $\psi_i + \alpha_i$ and the preferred direction ϕ_i toward the destination of flight f_i.

A. Optimization Problem Formulation

Our collision avoidance problem can be expressed as an optimization problem where the primary objective is to avoid conflicts, and if they occur to have conflicts as less severe as possible, and the secondary objective is to fly toward the destination. These prioritized objectives can be expressed very simply by the following objective function returning a tuple of costs:

$$f_{c,a}(x) = (N_c, C_{S_{\text{dist}}}(x), C_{S_{\text{time}}}(x), C_{angle}(x)) \tag{7}$$

where N_c is the number of conflict occurrences, $C_{S_{\text{dist}}}$ and $C_{S_{\text{time}}}(x)$ are the distance and time severity costs of conflicts, and $C_{angle}(x)$ is the angular deviation cost, as defined in the previous subsection.

This formulation allows us to prioritize our costs. When comparing two values of the objective function $f_{c,a}$, the costs will simply be compared in their lexicographic order. The subscripts c, a in $f_{c,a}$ are here to remind the reader that the objectives of this function are to avoid conflicts (c) or minimize their severity, and also to minimize the angular deviations (a), in this order of priority.

Implementing a centralized anti-collision logic will consist in solving, at each time step t, the following unconstrained optimization problem $\mathcal{P}(\mathcal{F}, \Delta, \tau, t)$:

$$(\mathcal{P}(\mathcal{F}, \Delta, \tau, t)) \quad : \quad \min_{x\in\mathcal{D}} \quad f_{c,a}(x) \tag{8}$$

where:

- $\mathcal{F}$ is the set of airborne aircraft or UAS flying in the considered geographic area at time t,
- Δ is the standard separation,
- τ is the anticipation horizon used to detect conflicts,
- $x = (\alpha_1, \dots, \alpha_n)^T$ is the vector of $n = |\mathcal{F}|$ candidate maneuvers (*i.e.* direction changes) used as decision variables,
- $\mathcal{D}$ is the domain of the angular maneuvers x determined by the maximum turning rate of each aircraft or UAS.

IV. A Centralized Collision Avoidance Algorithm

A. Algorithm Description

Algorithm 1 describes the proposed centralized anti-collision algorithm used in this study, with the sole purpose of studying the time horizon effect. At each time step, the algorithm updates the set $\mathcal{F}$ of flights present in the considered area using function UPDATEFLIGHTS, and solves the optimization problem $\mathcal{P}(\mathcal{F}, \Delta, \tau, t))$ described in the previous section, using the function SOLVEPROBLEM. This function returns a set of optimized maneuvers. The direction changes are then implemented by CHANGEDIRECTIONS.

Algorithm 1 Centralized Anti-collision Algorithm.

```
Require: $t_{start}$, $t_{end}$, time step $\delta_t$, and flight plans $Fp$ with initial
    positions and velocities, entry times, and flight destinations
1: $t \leftarrow t_{start}$
2: $\mathcal{F} \leftarrow \emptyset$                                    ▷ Active flights
3: $x_{\text{best}} \leftarrow$ None
4: while $t \le t_{end}$ do
5:     $\mathcal{F} \leftarrow$ UPDATEFLIGHTS($\mathcal{F}, F_p, t$)
6:     $x_{best} \leftarrow$ SOLVEPROBLEM($\mathcal{P}(\mathcal{F}, \Delta, \tau, t)$)
7:     $\mathcal{F} \leftarrow$ CHANGEDIRECTIONS($\mathcal{F}, x_{best}$)
8:     $t \leftarrow t + \delta t$
9: end while
```

In algorithm 1, we could directly use any suitable optimization algorithm $\mathcal{A}$ to implement SOLVEPROBLEM, considering all n flights present in the area at time t, and optimizing in the n-dimensional space of decision variables $x = (\alpha_1, \dots, \alpha_n)^T$. However, some of these flights might not be in conflict with any other at that time, or the conflicting pairs might be grouped into several separate clusters.

The optimization process can be greatly improved by considering these clusters – i.e. the connected components of the graph of conflicts – as sub-problems that can be optimized separately. The function SOLVEPROBLEM that implements this strategy is described in Algorithm 2.

Algorithm 2 Problem Solving

```
Require: An optimization algorithm $\mathcal{A}$
 1: function SOLVEPROBLEM($\mathcal{P}(\mathcal{F}, \Delta, \tau, t)$)
 2:     $x_{trial} \leftarrow$ TOWARDDESTINATION($\mathcal{F}$)    ▷ Initial candidate maneuvers, toward the destinations or as close as possible
 3:     $\mathcal{F}_{trial} \leftarrow$ CHANGEDIRECTIONS($\mathcal{F}, x_{trial}$)    ▷ Candidate flights with updated directions
 4:     $\mathcal{C} \leftarrow$ DETECTCONFLICTS($\mathcal{F}_{trial}$)
 5:     $\mathcal{L} \leftarrow$ MAKECLUSTERS($\mathcal{C}$)    ▷ Independent sub-problems (connected components of the conflict graph)
 6:     $\mathcal{L}_{old} \leftarrow \emptyset$
 7:     while $\mathcal{L} \neq \emptyset$ and $\mathcal{L} \neq \mathcal{L}_{old}$ do
 8:         $l \leftarrow$ SOLVESUBPROBLEMS($\mathcal{L}, \mathcal{A}$)    ▷ Solve each sub-problem independently, using algorithm $\mathcal{A}$
 9:         $x_{trial} \leftarrow$ CONCATSOLUTIONS($x_0, l$)    ▷ Concatenate the sub-problem solutions to form a candidate global solution
10:         $\mathcal{F}_{trial} \leftarrow$ CHANGEDIRECTIONS($\mathcal{F}, x_{trial}$)
11:         $\mathcal{C} \leftarrow$ DETECTCONFLICTS($\mathcal{F}_{trial}$)
12:         $\mathcal{L}_{old} \leftarrow \mathcal{L}$
13:         $\mathcal{L} \leftarrow$ UPDATECLUSTERS($\mathcal{C}, \mathcal{L}_{old}$)
14:     end while
15:     return $x_{trial}$
16: end function
```

In Algorithm 2, an initial vector of maneuvers x_{trial} is computed, maneuvering each flight toward its destination. Due to the maximum turning rate constraint, flying directly in the preferred direction – toward destination – might not be possible. In this case, the initial candidate maneuver aims at the feasible direction closest to preferred one. The set of flights $\mathcal{F}_{trial}$ returned by CHANGEDIRECTIONS is the same as $\mathcal{F}$, except that the velocities now follow the candidate directions.

The set $\mathcal{C}$ of conflicting pairs of flights is then computed (DETECTCONFLICTS), assuming the flights follow these candidate directions, and the clusters $\mathcal{L}$ are computed from the graph of conflicts (MAKECLUSTERS).

The function SOLVESUBPROBLEMS simply applies an optimization algorithm $\mathcal{A}$ to each cluster in $\mathcal{L}$ and returns the optimized maneuvers for each sub-problem. In this study, we have used a Differential Evolution algorithm, described in the next subsection IV-B. The new candidate solution x_{trial} for the global situation involving all flights is obtained by aggregating the partial solutions, using CONCATSOLUTIONS.

Solving the sub-problems in $\mathcal{L}$ independently is not sufficient to ensure that the new global candidate solution x_{trial} is without conflicts: We might have solved all conflicts within each cluster, but created new ones with flights from other clusters, or with flights that were initially not in conflict with any other.

In such cases, we have to reconsider our clusters (UPDATECLUSTERS), either by merging them, or by adding new flights, and we have to solve these new sub-problems, until there remain no conflicts in the overall situation ($\mathcal{C}$ and $\mathcal{L}$ are empty), or until the clusters are unchanged. This last case may occur when the optimal solution is not without conflicts. This process is implemented by the "while" loop in Algorithm 2, which checks if the clusters $\mathcal{L}$ after a solving iteration are different from the previous ones $\mathcal{L}_{old}$.

B. The Differential Evolution Algorithm

Let us now describe the Differential Evolution (DE) algorithm that we used to solve the anti-collision problems. Differential Evolution is a metaheuristic, proposed by R. Storn and K. Price in [39], for optimization problems with real-valued decision variables. We chose this algorithm for its simplicity and its efficiency on multi-modal objective functions.

The Differential Evolution algorithm applies a recombination and selection process to a population $\{x_1, \ldots, x_{popsize}\}$ of candidate solutions through a number of generations, as described in Algorithm 3, and returns the best element(s) of the population at the end. Each candidate solution x_i is a vector of dimension dim of floating-point values. The recombination operator RECOMBINE is described in Algorithm 4.

Algorithm 3 Differential Evolution Algorithm.

```
Require: Objective function $f$, problem dimension $dim$, population size $popsize$, maximum number of generations $maxiter$, crossing rate $CR$, amplification factor $F$
 1: $\{x_1, \ldots, x_{popsize}\} \leftarrow$ INITPOP($popsize, dim$)
 2: $iter \leftarrow 0$
 3: while $iter < maxiter$ do
 4:     for individual $i = 1$ to $popsize$ do
 5:         Randomly pick $x_a$,$x_b$, and $x_c$, with $a, b, c, i$ all different
 6:         $y_i \leftarrow$ RECOMBINE($x_i, x_a, x_b, x_c$)
 7:         $x_i \leftarrow y_i$ if $f(y_i) < f(x_i)$
 8:     end for
 9: end while
10: $x_{best} \leftarrow \underset{x \in \{x_1, \ldots, x_{popsize}\}}{\arg\max} f(x)$
11: return $x_{best}$
```

Algorithm 4 Recombination Operator of the Differential Evolution Algorithm.

```
Require: Dimension $dim$, crossing rate $CR$, amplification factor $F$
 1: function RECOMBINE($x_i, x_a, x_b, x_c$)
 2:     $y_i \leftarrow x_i$
 3:     $r \leftarrow$ RANDOMINT($\{1, \ldots, dim\}$)
 4:     for $j = 1$ to $dim$ do
 5:         $r_f \leftarrow$ RANDOMFLOAT($[0, 1]$)
 6:         if $j = r$ or $r_f < CR$ then
 7:             $y_{i,j} \leftarrow x_{a,j} + F \times (x_{b,j} - x_{c,j})$
 8:         end if
 9:     end for
10:     return $y_i$
11: end function
```

C. Applying Differential Evolution to the Collision Avoidance Problem

Applying the DE algorithm to our anti-collision problem is straightforward. We can use it directly as the algorithm $\mathcal{A}$ required in Algorithm 2.

The dimension of the problem is $dim = n_c$, where n_c is size of the cluster of flights in the sub-problem to solve. The population is initialized by building $popsize$ vectors of maneuvers. Each vector is built by selecting random maneuvers (*i.e.* direction changes) for the n_c flights, with a uniform distribution in the angular domain defined by the maximum turning rate of each flight.

V. The Time Horizon Effect – Case Study with Two Flights

Let us now consider two flights entering a circular geographic area of radius $100\,\mathrm{NM}$ (nautical miles) at the same time, both flying at $250\,\mathrm{kts}$ (knots) and set on courses converging at an angle of 30 degrees. Figure 2 shows the trajectories of these two flights when no collision avoidance algorithm is applied.

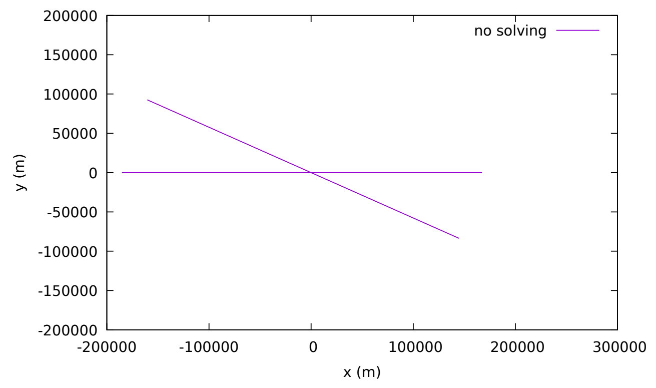

Figure 2: Initial trajectories of two flights converging at an angle $30°$ at $250\,\mathrm{kts}$, with start and end points on a circle of radius $100\,\mathrm{NM}$.

A. Illustration of the Time Horizon Effect

Figure 3 shows the trajectories for the same flights when Algorithm 2 is applied with a parameter $\eta = 2$ for the angular deviation cost (see Def. III.7), a population size of 30 for the Differential Evolution, and parameters $F = 7$, $CR = 0.1$, and $maxiter = 2000$. On Fig. 3 the simulated flight time was limited to 1 hour. The time horizon τ for the conflict detection is set to 10 minutes. We can observe that the two flights are locked on parallel courses and that none of them is able to reach its destination.

Such a situation is clearly an issue, both in terms of efficiency and safety: The anti-collision algorithm does ensure separation, but the flights never reach their respective destinations. In real life, they would eventually run out of power (or fuel, depending on the type of propulsion). This is what would occur in the situation shown on Figure 4 which shows the same traffic scenario, but with a flight time extended to 6 hours.

B. Causes of the Time Horizon Effect

The time horizon effect illustrated on Figs. 3 and 4 was first identified, to our knowledge, by Durand in [6], [7] when applying the distributed geometric algorithm ORCA to constant-speed UAS. One could think that this effect might be caused by the distributed nature of the D&A logic, and by the lack of knowledge of the others intents when choosing a lateral maneuver. This is not the case.

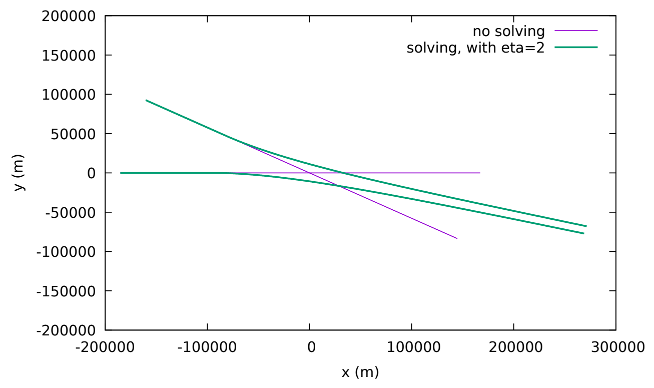

Figure 3: Anti-collision solution for two flights converging at an angle $30°$ at $250\,\mathrm{kts}$, with parameter $\eta = 2$. One hour of flight was simulated.

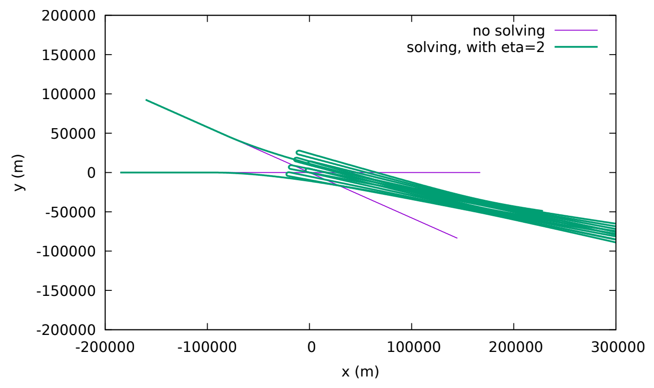

Figure 4: Anti-collision solution for two flights converging at an angle $30°$ at $250\,\mathrm{kts}$, with parameter $\eta = 2$. Six hours of flight were simulated.

From the examples Figs. 3 and 4, where we used a centralized anti-collision algorithm with full knowledge of all the flights, we can see that the time horizon effect has nothing to do with the distributed nature of most D&A logics.

Actually, for any distributed or centralized algorithm for which conflict resolution is the top priority, it might happen that the easiest way to solve a conflict occurring within a finite time horizon is to postpone the trajectories' crossing beyond the time horizon, by maneuvering toward parallel tracks. This pathological situation is more likely to occur when the conflicting aircraft or UAS are flying at similar speeds and can only make slow and limited velocity changes. This is typically the case for commercial flights flying at high altitudes, and it might also be the case for some Urban Aerial Mobility vehicles cruising in lower airspace in the future, or for some existing fixed-wing UAS.

Ultimately, the true reason why the time horizon effect occurs is that we use a myopic resolution strategy where the actions taken at a given time step do not take account of all the consequences on the future time steps. Even though we use a global optimization algorithm to solve conflicts, the whole optimization process over several time steps can be considered as a greedy heuristic: the algorithm selects the maneuvers that yield the best results at the current time step, without planning beyond the horizon of the conflict detection.

C. Mitigating the Horizon Effect with Inequitable Maneuvers

Symmetry is clearly an aggravating factor for the horizon effect, and any action that breaks the symmetries in a conflicting situation is likely to help. Unfortunately, in many D&A logics such as MVP [25], or ORCA [35] and its variants (*e.g. [7]*, the maneuvering effort is shared among the flights. It is usually preferred to have two flights making small maneuvers instead of only one making a bigger one.

In our centralized algorithm, we can implement this equity objective by choosing a parameter value $\eta > 1$ in the cost of the angular deviations (see Def.III.7). For example, with $\eta = 2$, the cost of deviating both of two flights of 1 will be $1^2 + 1^2 = 2$, whereas this cost will be $2^2 = 4$ if only one is deviated of 2 and the other is not deviated. Consequently, the optimization algorithm will guide the search toward solutions where the maneuvers are shared among the flights, as illustrated on Fig. 3 in section V-A.

At the opposite, choosing $\eta < 1$ will favor inequitable maneuvers. For example, with $\eta = 0.9$, the cost of two deviations of 1 each is still 2 ($1^{0.9} + 1^{0.9}$) but it is now greater than the cost of a single deviation of value 2, which is $2^{0.9} < 2$, so the optimization algorithm will favor solutions deviating only one of the two flights.

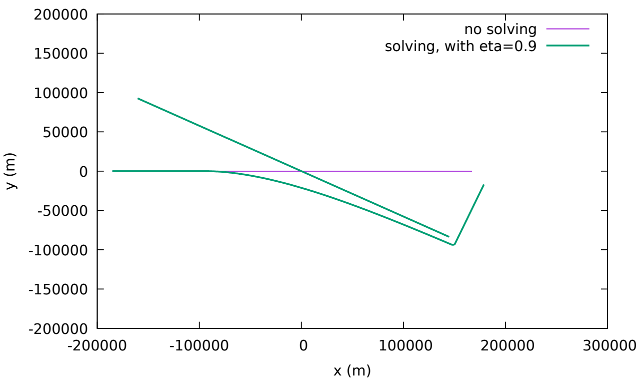

Figure 5: Anti-collision solution for two flights converging at an angle 30° at 250 kts, with parameter $\eta = 0.9$.

Figure 5 shows the result obtained with $\eta = 0.9$ for the same two flights as in section V-A. Only one of the two flights is maneuvered to avoid collision. It still has to follow a parallel track until the other reaches its destination. The deviated flight can then resume its navigation toward its own destination.

Favoring inequitable maneuvers with $\eta < 1$ does not completely solve the parallel track issue, but at least both flights end up at their respective destinations, which was not the case with equitable maneuvers ($\eta = 2$)

In a previous work [8], we proposed a distributed geometric algorithm – the Dual-Horizon Reciprocal Collision-Avoidance (DH-ORCA) algorithm – where the shorter time horizon is used to enforce separation, and the broader horizon is used to help cross the trajectories when necessary. A similar strategy could be used in the centralized algorithm proposed here.

Whatever the chosen mitigating strategy, the important point to emphasize here is that a D&A logic that prioritizes collision avoidance over all other objectives does not solve all issues and does not by itself guarantee the safety of the overall system.

VI. Results on Random Traffic Scenarios

A. Traffic Scenarios

In this section, we use dense random scenarios (approx. 8.5 flight/10 000 NM^2) that are specifically designed to be more likely to exhibit the time horizon effect (constant speeds, lateral maneuvers only). The objective here is not to simulate realistic environments, but to test the limits of the collision avoidance algorithm.

In our scenarios, all UAS fly at constant speeds uniformly drawn between 230 and 270 knots, at the same altitude. We consider only lateral maneuvers to avoid collisions, with a maximum turning rate of 3 degrees per second. The time horizon for the conflict detection is set to $\tau = 10$ minutes.

The flight entry times are randomly generated following a Poisson distribution having an average incoming flow of 60 entries per hour. Once the entry times are selected, the entry point of each incoming flight is randomly chosen within the available domain on a circle of radius 150 nautical miles, considering protection zones around the previous flights, and assuming these flights follow direct routes toward their destinations. The entry point selection is illustrated in Figure 6. These exclusion zones of radius $\Delta_{entry} = 4 \times \Delta$ around the other flights are here to avoid separation losses at entry, and to give room for the entering flights to maneuver.

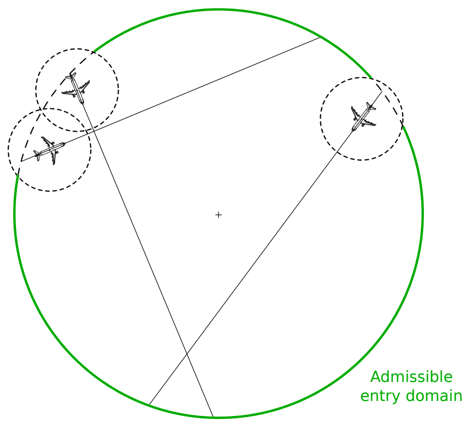

Figure 6: Entry point selection in the admissible domain, considering exclusion zones defined around previous flights.

The route of each flight is then randomly chosen as illustrated on Fig. 7 by drawing in a uniform distribution $[-\alpha, \alpha]$, with $\alpha = 70°$. The exit point is then simply the intersection of the route with the circular boundary.

These scenarios are designed so that no loss of separation occurs at entry if all UAS fly directly toward their destination. When a D&A algorithm modifies the trajectories to avoid collisions, these modified trajectories might interfere with entering flights. In such cases, the entering flights are delayed until they can enter while respecting the exclusion zones around the other UAS.

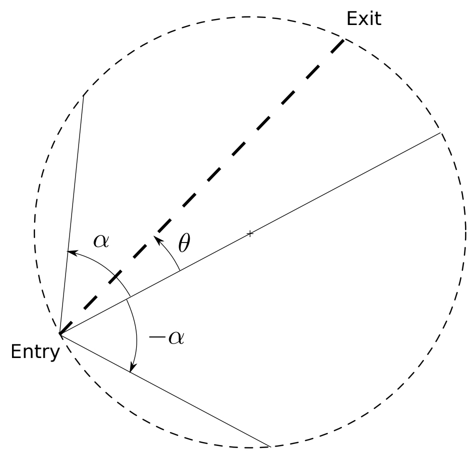

Figure 7: .

B. Results with $\eta = 2$ (shared maneuvers)

Figure 8 shows the trajectories obtained by applying Algorithm 2 every 5 seconds when simulating 6 hours of the traffic scenario described in section VI-A. The parameter η of Eq. (6) in definition III.7 of the angular deviation cost is here set to $\eta = 2$. With this value of $\eta > 1$, the optimization algorithm will favor solutions where maneuvers are shared among the flights.

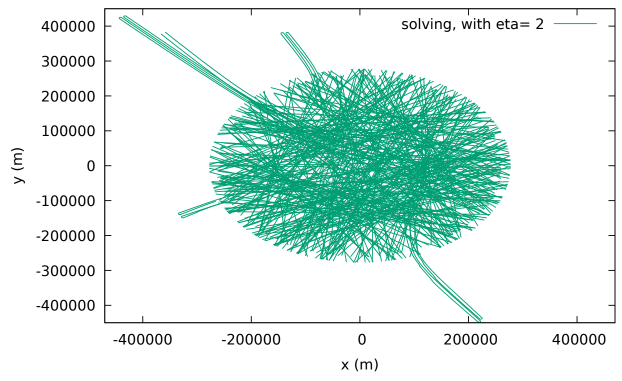

Figure 8: Anti-collision solution for 6 hours of dense traffic scenario with an average incoming flow of 60 flight per hour, with parameter $\eta = 2$.

The Differential Evolution parameters are the same as in section V, except for $maxiter$, the maximum number of iterations, which was set to 20000, to be on the safe side. A more extensive parameter study would be required to tune the parameter values. The 6 hours of traffic were simulated in 6236 seconds of CPU time (approximately 1 hour 50 minutes) on an `Intel(R) Core(TM) i9-9940X CPU @ 3.30GHz`.

In this dense scenario, 65 separation losses between flights were observed, with a severity loss of 0.04 (*i.e.* a 4% infringement of the standard separation, at most). The total duration of the separation losses is 1240 seconds. The total delay for all flights is 315 seconds.

We can see on Fig. 8 that several flights are subject to the pathological time horizon effect, with trajectories locked on parallel tracks and missing their destination. The total time spent locked on parallel tracks is 31185 seconds, from which 10905 were spent flying away from the destination. The mathematical expressions for these parallel track metrics can be found in [8].

C. Results with $\eta = 0.9$ (inequitable maneuvers)

With $\eta = 0.9$, the optimization algorithm assigns larger lateral maneuvers to a smaller number of flights than with $\eta = 2$, and it should be slightly less subject to the parallel track issue.

Figure 9 shows the trajectories obtained on the same scenario as in section VI-B. We can see that there are no parallel tracks going away from the flights' destinations.

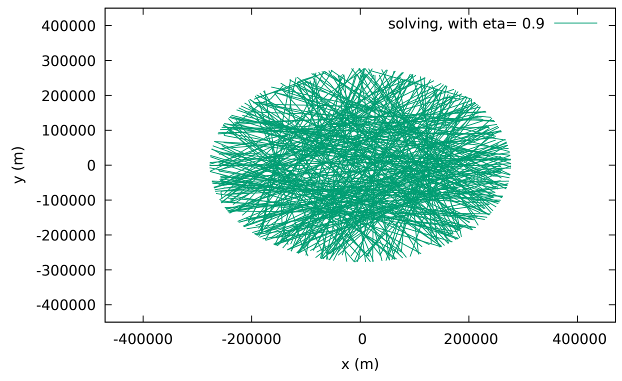

Figure 9: Anti-collision solution for 6 hours of dense traffic scenario with an average incoming flow of 60 flights per hour, with parameter $\eta = 0.9$.

The number of separation losses between flights is 48, with a severity of 0.03, and a total duration of 770 seconds. The total delay is 965 seconds. There are still some parallel tracks, with a total time 2225 seconds. This is certainly due to situations similar to the one shown on Fig. 5, where one UAS flies directly toward its destination and the other has to follow a parallel track until it can safely turn toward its own destination. However, as opposed to the results obtained with $\eta = 2$, the time spent on parallel tracks and flying away from the destination is reduced to 0.

Table I summarizes the results on the dense traffic scenario, when sharing the maneuvers ($\eta = 2$), or when inequitable maneuvers are preferred ($\eta = 0.9$).

	$\eta = 2$	$\eta = 0.9$
LoS	65	48
LoS severity	0.04	0.03
LoS duration (s)	1240	770
Total delay (s)	315	965
Parallel tracks duration (s)	31185	2225
Parallel and away (s)	10905	0

TABLE I. Results of the centralized collision avoidance algorithm, with $\eta = 2$ (equitable maneuvers) and $\eta = 0.9$ (inequitable maneuvers), on a random scenario of six hours of dense traffic – LoS= Loss of Separation.

VII. Conclusion

In this paper, we have introduced a centralized collision avoidance algorithm based on Differential Evolution, for the purpose of studying the time horizon effect that was identified and studied in [6], [7] in a decentralized context.

We have shown that a centralized algorithm with full knowledge of the flights' intents can also be plagued by this effect where trajectory crossings might be postponed beyond the detection horizon, and where some flights might end up on parallel tracks, missing their destination.

The primary cause of the time horizon effect is not the distributed nature of most D&A logics. There are probably several factors favoring the apparition of this undesirable effect. One of them is the symmetry in some traffic situations, and in such cases we have seen that sharing the maneuver among the flights might not be a good idea. Another cause is that conflicts are detected within a finite time horizon and that the D&A logic prioritizes conflict resolutions over the deviation costs. With this prioritization, the effect occurs because of the myopic strategy that consists in selecting the best action at the current time step – *i.e.* the best set of direction changes that solves the collision avoidance problem – without considering the consequences on the future time steps beyond the detection horizon. In a sense, although we are applying a global optimization algorithm at every time step, we actually use a greedy heuristic when considering the path finding problem over sequences of successive time steps.

By testing our centralized approach on dense random scenarios, we have shown that sharing the maneuver effort among the flights is an aggravating factor. Favoring inequitable maneuvers breaks the symmetry that may occur in pathological situations. It actually improves the results to a certain extent: Although all flights reach their respective destinations, there still remain many pathological parallel tracks situations.

A more promising strategy would be to plan trajectory crossings beyond the time horizon of the anti-collision logic. The use of dual-horizon logics such as the one proposed in [8] in a decentralized context can help mitigate the horizon effect. Such approaches rely on two nested logics: one with the longer time horizon which aims at crossing the trajectories when intended so, and the other with the shorter time horizon devoted to collision avoidance. In future works, we plan to study such a dual-horizon approach in a centralized context, and to compare distributed approaches with centralized ones, implementing efficient dual-horizon strategies in both contexts.

The most important lesson that may be learned from the current study is that prioritizing collision avoidance over all other objectives does not guarantee the safety of the overall UAS Traffic Management (UTM) system. The certification process of the UAS D&A logics and of the UTM system as a whole should not only focus on the conditions in which separation losses may occur, but also on the conditions in which the horizon effect may happen.

Acknowledgment

We would like to thank Valentin Latour. The brainstorming sessions we had with him during his summer internship in our team motivated us to produce the work presented in this paper.

References

[1] Vishwanath Bulusu, Raja Sengupta, and Zhilong Liu. Unmanned aviation: To be free or not to be free? a complexity based approach. In *7th International Conference on Research in Air Transportation*, ICRAT 2016 Proceedings, Philadelphia, PA, June 2016. Drexel University.

[2] Parimal Kopardekar, Joseph Rios, Thomas Prevot, Marcus Johnson, Jaewoo Jung, and John E Robinson. Unmanned aircraft system traffic management (utm) concept of operations. In *AIAA Aviation and Aeronautics Forum (Aviation 2016)*, number ARC-E-DAA-TN32838, 2016.

[3] Richard Alligier, Cyril Allignol, Nicolas Barnier, Nicolas Durand, and Ruixin Wang. Detect and avoid algorithm for uas with 3d-maneuvers. In *ICRAT 2018, 8th International Conference on Research in Air Transportation*, 2018.

[4] Konstantinos Dalamagkidis, Kimon P. Valavanis, and Les A. Piegl. *On integrating unmanned aircraft systems into the national airspace system: issues, challenges, operational restrictions, certification, and recommendations*. Springer, 2009.

[5] Reece A. Clothier, Brendan P. Williams, and Neale L. Fulton. Structuring the safety case for unmanned aircraft system operations in non-segregated airspace. *Safety science*, 79:213–228, 2015.

[6] Nicolas Durand and Nicolas Barnier. Does atm need centralized coordination? autonomous conflict resolution analysis in a constrained speed environment. In *ATM seminar 2015, 11th USA/EUROPE Air Traffic Management R&D Seminar*, 2015.

[7] Nicolas Durand. Constant speed optimal reciprocal collision avoidance. *Transportation research part C: emerging technologies*, 96:366–379, 2018.

[8] Richard Alligier, David Gianazza, Nicolas Durand, and Xavier Olive. Dual-horizon reciprocal collision avoidance for aircraft and unmanned aerial systems. *Journal of Intelligent & Robotic Systems*, 107(1):1–24, 2023.

[9] International Civil Aviation Organization. *Airborne Collision Avoidance System (ACAS) Manual*, 2006. Doc 9863, AN/461.

[10] Mykel J. Kochenderfer and James P. Chryssanthacopoulos. Robust airborne collision avoidance through dynamic programming. *Massachusetts Institute of Technology, Lincoln Laboratory, Project Report ATC-371*, 130, 2011.

[11] Sheng Li, Maxim Egorov, and Mykel Kochenderfer. Optimizing collision avoidance in dense airspace using deep reinforcement learning. *arXiv preprint arXiv:1912.10146*, 2019.

[12] Kyle D. Julian, Jessica Lopez, Jeffrey S. Brush, Michael P. Owen, and Mykel J. Kochenderfer. Policy compression for aircraft collision avoidance systems. In *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pages 1–10. IEEE, 2016.

[13] Iyad Lahsen-Cherif, Huan Liu, and Catherine Lamy-Bergot. Real-time drone anti-collision avoidance systems: an edge artificial intelligence application. In *2022 IEEE Radar Conference (RadarConf22)*, pages 1–6. IEEE, 2022.

[14] Nicolas Durand, Jean-Marc Alliot, and Joseph Noailles. Automatic aircraft conflict resolution using genetic algorithms. In *11th Annual Symposium on Applied Computing*, pages 289–298, Philadelphia, PA, February 1996. ACM.

[15] Nicolas Durand and Jean-Marc Alliot. Optimal resolution of en route conflicts. In *Proceedings of the 1st ATM R&D Seminar*, Saclay, France, June 1997.

[16] Lucia Pallottino, Antonio Bicchi, and Éric Feron. Mixed integer programming for aircraft conflict resolution. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, Montréal, Canada, August 2001.

[17] Maxime Gariel and Éric Feron. 3d conflict avoidance under uncertainties. In *28th Digital Avionics Systems Conference*, DASC 2009 Proceedings, pages 4.E.3–1–4.E.3–8, Orlando, FL, October 2009. AIAA, IEEE.

[18] Thibault Lehouillier, Jérémy Omer, François Soumis, and Guy Desaulniers. Two decomposition algorithms for solving a minimum weight maximum clique model for the air conflict resolution problem. *European Journal of Operational Research*, 256(3):696 – 712, 2017.

[19] Jérémy Omer and Farges Jean-Loup. Hybridization of nonlinear and mixed-integer linear programming for aircraft separation with trajectory recovery. *IEEE Transactions on Intelligent Transportation Systems*, 14(3):1218–1230, Sept 2013.

[20] Cyril Allignol, Nicolas Barnier, Nicolas Durand, and Jean-Marc Alliot. A new framework for solving en-route conflicts. In *Proceedings of the 10th ATM R&D Seminar*, Chicago, IL, June 2013.

[21] Ruixin Wang, Richard Alligier, Cyril Allignol, Nicolas Barnier, Nicolas Durand, and Alexandre Gondran. Cooperation of combinatorial solvers for en-route conflict resolution. *Transportation research. Part C, Emerging technologies*, 114:36–58, May 2020.

[22] Karim Zeghal. A comparison of different approaches based on force fields for coordination among multiple mobiles. In *Proceedings. IEEE/RSJ International Conference on Intelligent Robots and Systems. Innovations in Theory, Practice and Applications*, volume 1, pages 273–278, October 1998.

[23] Jana Košecká, Claire Tomlin, Georges J. Pappas, and Shankar Sastry. 2 1/2 d conflict resolution maneuvers for atms. In *Proceedings of the 37th IEEE Conference on Decision and Control*, volume 3, pages 2650–2655, December 1998.

[24] Martin S. Eby and Wallace E. Kelly, III. Free flight separation assurance using distributed algorithms. In *IEEE Aerospace Conference. Proceedings*, volume 2, pages 429–441, March 1999.

[25] Jacco M. Hoekstra, Ronald N.H.W Van Gent, and Rob R.C.J Ruigrok. Designing for safety: the 'free flight' air traffic management concept. *Reliability Engineering & System Safety*, 75(2):215 – 232, 2002.

[26] Rob C.J. Ruigrok and Jacco M. Hoekstra. Human factors evaluations of free flight: Issues solved and issues remaining. *Applied Ergonomics*, 38(4):437 – 455, 2007. Flightdeck of the Future.

[27] Baraa M. Albaker and Nasrudin A. Rahim. A survey of collision avoidance approaches for unmanned aerial vehicles. In *2009 international conference for technical postgraduates (TECHPOS)*, pages 1–7. IEEE, 2009.

[28] Marc Brittain and Peng Wei. Autonomous separation assurance in an high-density en route sector: A deep multi-agent reinforcement learning approach. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3256–3262, 2019.

[29] Marc W Brittain and Peng Wei. One to any: Distributed conflict resolution with deep multi-agent reinforcement learning and long short-term memory. In *AIAA Scitech 2021 Forum*, page 1952, 2021.

[30] Ramon Dalmau and Eric Allard. Air traffic control using message passing neural networks and multi-agent reinforcement learning. *Proceedings of the 10th SESAR Innovation Days, Virtual Event*, pages 7–10, 2020.

[31] Calin Andrei Badea, Jan Groot, Andres Morfin Veytia, Marta Ribeiro, Joost Ellerbroek, Jacco Hoekstra, and Ramon Dalmau. Lateral and vertical air traffic control under uncertainty using reinforcement learning. In *12th SESAR Innovation Days*, 2022.

[32] Egidio D'Amato, Massimiliano Mattei, and Immacolata Notaro. Distributed reactive model predictive control for collision avoidance of unmanned aerial vehicles in civil airspace. *Journal of Intelligent & Robotic Systems*, 97(1):185–203, 2020.

[33] Paolo Fiorini and Zvi Shiller. Motion planning in dynamic environments using velocity obstacles. *The international journal of robotics research*, 17(7):760–772, 1998.

[34] Jur Van den Berg, Ming Lin, and Dinesh Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *2008 IEEE international conference on robotics and automation*, pages 1928–1935. Ieee, 2008.

[35] Jur Van Den Berg, Stephen J Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. In *Robotics Research: The 14th International Symposium ISRR*, pages 3–19. Springer, 2011.

[36] Suthes Balasooriyan. Multi-aircraft conflict resolution using velocity obstacles. Master's thesis, TUDelft, 2017.

[37] Jurrian G. d'Engelbronner, Clark Borst, Joost Ellerbroek, Marinus M. Van Paassen, and Max Mulder. Solution-space–based analysis of dynamic air traffic controller workload. *Journal of Aircraft*, 52(4):1146–1160, 2015.

[38] Jamie Snape and Dinesh Manocha. Navigating multiple simple-airplanes in 3D workspace. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3974–3980, Anchorage, AK, 2010.

[39] Rainer Storn and Kenneth Price. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341, 1997.