Dual-Horizon Reciprocal Collision Avoidance for Aircraft and Unmanned Aerial Systems

Richard Alligier, David Gianazza, Nicolas Durand, Xavier Olive

Dual-Horizon Reciprocal Collision Avoidance for Aircraft and Unmanned Aerial Systems

Richard Alligier[1*], David Gianazza[1], Nicolas Durand[1] and Xavier Olive[2]

[1*]ENAC, Université de Toulouse, France.
[2]ONERA DTIS, Université de Toulouse, France.

*Corresponding author(s). E-mail(s): richard.alligier@enac.fr;
Contributing authors: david.gianazza@enac.fr; nicolas.durand@enac.fr;
xavier.oliver@onera.fr;

Abstract

The aircraft conflict detection and resolution problem has been addressed with a wide range of centralised methods in the past few decades, e.g. constraint programming, mathematical programming or metaheuristics. In the context of autonomous, decentralized collision avoidance without explicit coordination, geometric methods provide an elegant, cost-effective approach to avoid collisions between mobile agents, provided they all share a same logic and a same view of the traffic. The Optimal Reciprocal Collision Avoidance (ORCA) algorithm is a state-of-the art geometric method for robot collision avoidance, which can be used as a Detect & Avoid logic on-board aircraft or Unmanned Aerial Vehicles. However, ORCA does not handle well some degenerate situations where agents operate at constant or near-constant speeds, which is a widespread feature of commercial aircraft or fixed-winged Unmanned Airborne Systems. In such degenerate situations, pairs of aircraft could end up flying parallel tracks without ever crossing paths to reach their respective destination. The Constant Speed ORCA (CS-ORCA) was proposed in 2018 to better handle these situations. In this paper, we discuss the limitations of both ORCA and CS-ORCA, and introduce the Dual-Horizon ORCA (DH-ORCA) algorithm, where two time horizons are used respectively for short-term collision avoidance and medium-term path-crossing. We show that this new approach mitigates the main issues of ORCA and CS-ORCA and yields better performances with dense traffic scenarios.

Keywords: Collision Avoidance, aircraft conflict resolution, self-separation, Optimal Reciprocal Collision Avoidance, Air Traffic Control, Unmanned Airborne Systems

1 Introduction

Automated conflict detection and resolution is a topic of major interest in the field of air traffic management (Allignol et al (2012)). The problem has been largely addressed from a centralized or decentralized point of view, using deterministic or stochastic algorithms and considering the problem from a strategical or tactical point of view.

Current research and advances around Urban Air Mobility (UAM), Unmanned Air Vehicles (UAV) and more generally Unmanned Airborne Systems (UAS) flying in the lower airspace lead researchers to study collision avoidance algorithms in order to separate UAS. The expected emergence in a near future of a large-scale, dense UAS traffic brings the focus on short-horizon self-separation methods in a multi-agent context.

Collision avoidance has been widely studied in the context of robots trajectory planning. Among the most efficient algorithms for multi-agent collision avoidance, the Optimal Reciprocal Collision Avoidance (ORCA) introduced (van den Berg et al (2011)) relies on a geometric approach where each agent modifies its velocity vector so that the relative velocity with any other agent falls outside the region where collision would occur in a given time horizon. This geometric approach is designed to guarantee conflict resolution for the next minutes in an autonomous manner, without explicit coordination among the agents. Provided all agents apply the same logic for the geometric algorithm and share the same view of the traffic situation (positions and velocities of the other agents), the maneuvers computed independently by each agent are implicitly coordinated.

Note that the term "optimal" in ORCA relates to the fact that the respective sets of collision-avoiding velocities from which two conflicting robots choose their optimal respective velocities are reciprocally-avoiding and maximal (see definition 1 in van den Berg et al (2011)): one cannot find a larger couple of velocity sets that is reciprocally-avoiding. With ORCA, the smallest change in the relative velocity is chosen so as to prevent collision in a given time horizon, and this relative velocity change is fairly shared among the two agents. However, ORCA does not provide optimal solutions in terms of overall trajectory deviations. In fact, it can be considered as a greedy heuristic: The relative velocity of each conflicting pair is chosen optimally at each time step in the set of conflict-free velocities, but without trying to minimize trajectory deviations over a whole time interval.

Although very efficient for robots which can modify their direction and speed at will, Durand (2018) showed that the ORCA algorithm has limitations for agents such as fixed-wing UAVs or aircraft that can only move at constant or nearly-constant speeds. The Constant-Speed Optimal Reciprocal Collision Avoidance (CS-ORCA) proposed by Durand showed improvements over the ORCA logic, with fewer pairs of flights ending on parallel paths without ever reaching their destination.

In this paper, we discuss the limitations of CS-ORCA and propose a further improvement on the ORCA and CS-ORCA algorithms. We introduce the Dual-Horizon Reciprocal Collision Avoidance (DH-ORCA) algorithm where a short-horizon collision avoidance logic is complemented by a longer-horizon logic that helps agents to cross path with surrounding agents. We show that this dual-horizon logic provides improvements over the ORCA and CS-ORCA algorithms and has a significant impact on the residual separation losses and other safety and efficiency metrics.

We may expect that algorithms such as DH-ORCA could help tackle the problem of UAV Traffic Management in a safe, decentralized, autonomous way, even with the very large amounts of traffic expected in the near future. Such geometric algorithms are good candidates for possible airborne self-separation systems on-board commercial aircraft, or as the "Detect & Avoid" (D&A) logic envisioned for future Unmanned Aerial Systems.

After a short literature review on conflict detection and resolution methods in Section 2, we briefly discuss the context and the objectives of our study in Section 3. We then describe the ORCA algorithm in Section 4. The CS-ORCA variant is presented in Section 5, where its limitations are also discussed. Section 6 introduces the new dual-horizon method DH-ORCA. The experiment setup and traffic scenarios are presented in Section 7. Finally, the results are presented in Section 8, before the conclusion.

2 Literature Review

En-route conflict detection and resolution is currently managed by air traffic controllers who have a global view of traffic. They use a horizontal 2D representation of the traffic (a radar screen) and give different types of maneuvers (e.g. heading or flight level changes, small speed modifications) to maintain a minimum standard separation between aircraft. It has been known since the 90s that this task is highly combinatorial (see Granger and Durand (2003) for a discussion on the subject). When more than two aircraft are involved, it is more efficient to consider the global situations than to solve conflicts by pairs. However, when the number of aircraft increases, a global approach may become too complex and autonomous algorithms can become a good alternative. Automatic conflict resolution methods can thus be divided in two categories: on one hand centralized

approaches consider the whole situation and give instructions to every aircraft, on the other, autonomous approaches delegate the separation task to the different aircraft.

2.1 Centralized Approaches

Two types of centralized approaches can be found in the literature. In the first one, greedy methods (Krella et al (1989); Chiang et al (1997); Hu et al (2002)) use sequential algorithms to optimize trajectories one by one after ranking aircraft. The main challenge is then to find an appropriate ordering (Archambault and Durand (2004)). The second type of centralized approach considers the traffic situation as a global problem and tries to find a solution without prioritizing aircraft, using global optimization techniques such as evolutionary algorithms, mathematical programming, constraint programming, and making various assumptions on the traffic and/or the aircraft performances.

Durand et al (1996) showed that a genetic algorithm could efficiently solve multiple aircraft conflicts with simple maneuvers on real traffic data. The algorithm is all the more efficient when uncertainties on trajectory prediction can be reduced (Alliot et al (2001)). Other approaches used mathematical methods (semi-definite programming) (Oh et al (1997); Frazzoli et al (2001)), but solutions were only locally optimal, and the model required perfect trajectory prediction which is unrealistic. A perfect trajectory prediction was also used by Pallottino et al (2001, 2002) in the early 2000s in a mathematical model using Mixed Integer Linear Programming, which could be solved by CPLEX and ensure the global optimality of the solution. Christidoulou extended the model in 3D, but still required constant speed during climbing phases (Christodoulou and Kontogeorgou (2008)). An improved version adding uncertainties to the trajectory headings was introduced by Gariel and Feron (2009) in 2009 but all maneuvers needed to be executed at the same time. Omer and Farges (2013) introduced in the early 2010s a Mixed-Integer Linear Programming approach taking into account the trajectory recovery. Alonso-Ayuso et al (2016) used a Mixed Integer Non Linear Optimization model that also took the trajectory recovery into account. Peyronne et al (2015) introduced a trajectory model using unrealistic B-Splines trajectories and a semi-infinite programming formulation of the constraints. Rey and Hijazi (2017) recently proposed a new complex number formulation and convex relaxations for the centralized problem.

Allignol et al (2013) proposed to separate the model from the resolution algorithm. For each aircraft, a number of alternative trajectories and a matrix of pairwise conflicts were pre-calculated, taking various uncertainties into account, before the optimization process was performed. A similar approach was proposed by Lehouillier et al (2017). In Wang et al (2020) different resolution methods were compared on the benchmark proposed in Allignol et al (2013).

2.2 Autonomous Approaches

Autonomous approaches appeared in the 90s with the Free Flight debate motivated by Air Traffic Control cost reduction. The development of Unmanned Airborne Systems in the lower airspace recently gave a new interest to these approaches because the high traffic demand in the lower airspace (Bulusu et al (2016)) might create dense situations that centralized approaches might not be able to handle, or at a greater cost. Sliding forces to coordinate maneuvers between aircraft (Zeghal (1998)) were one of the first autonomous approaches proposed in the late 90s. Potential or vortex fields (Košecká et al (1998)) as well as a model based on an analogy with electrical particle repulsion (Eby and Kelly (1999)) were also used. These algorithms took into account the aircraft speed constraints[1]. For situations involving many aircraft, these approaches relied on the principle that forces or potential fields virtually generated by aircraft on each other would add up. There is no guarantee that this principle always leads to a conflict free solution. Eby and Kelly (1999) inspired the Airborne Separation Assurance System (ASAS) approach developed by Hoekstra et al (2002). ASAS used a Modified Voltage Potential (MVP) close to Eby's definition. It was tested in different contexts, such as the Mediterranean Free-Flight (Ruigrok and Hoekstra (2007)). More recently, it was used to model airspace stability

[1]When cruising, an aircraft can easily change its direction while respecting a maximum turning rate, but it can only slightly modify its speed, within a narrow speed range.

and capacity in a decentralized separation context (Sunil et al (2017)) and in a Capacity Assessment Model (Emmanuel Sunil and Hoekstra (2018)). A geometric optimization approach solving complex situations by using an iterative process was proposed by Bilimoria (2000) and used in the Future Air traffic management Concepts Evaluation Tool (FACET) project (D. Bilimoria et al (2001)). The Free-flight Autonomous and Coordinated Embarked Solver (Granger et al (2001b,a)) used a token allocation strategy to coordinate sequential maneuvers. Le Ny and Pappas (2010) also used a geometric approach for scheduling crossing times of aircraft through a metering fix. Pallottino et al (2007) proposed a model where coordination is ensured by protected stacking areas in which each aircraft can move without conflicting with other aircraft. The areas are headed to the flight destination. Schouwenaars and Feron (2004) used a similar approach to plan safe trajectories in a decentralized way.

Some approaches use a set of rules to coordinate the conflict resolution of multiple aircraft. For instance, D'Amato et al (2020) use the *Right of Way rules*, as prescribed by the International Civil Aviation Organization (ICAO) for human piloted flights. This *Right of Way rules* creates constraints for each UAV. Then, considering these constraints, each UAV optimizes its own trajectory using constrained Model Predictive Control.

Geometric approaches using an implicit automated coordination were first introduced by van den Berg et al (2008) with the Reciprocal Velocity Obstacles and then with the Optimal Reciprocal Collision Avoidance (ORCA) (van den Berg et al (2011)) algorithm used in the present paper. ORCA was initially built to simulate robots using self-separation logic in a 2D environment. In air traffic control, Snape and Manocha (2010) have extended the model to the vertical dimension without segregating the horizontal and vertical maneuvers. More recently, geometric approaches have led to the definition of Solution Space Diagrams (SSD) that were used to analyze the dynamic of air traffic controller workload (d'Engelbronner et al (2015)) and were recently compared to the Modified Voltage Potential approach (Balasooriyan (2017)) showing that the latter seemed to be more efficient in many situations. As opposed to ORCA which provides coordinated maneuvers for agents actively trying to avoid each other, SSD – which is actually a Velocity Obstacle (VO) algorithm with speed and turning rate constraints – does not provide coordinated maneuvers. In ORCA, a modification of the relative speed is computed for the considered pair of aircraft using a Reciprocal Velocity Obstacle (RVO). This relative speed modification is then split equally between the involved aircraft. In the Variable Responsibility ORCA (VR-ORCA) (Guo et al (2021)) the required relative speed modification is not shared equally, the relative speed modification is split between the two aircraft in order to minimize a cost function that takes into account all the aircraft visible by both aircraft.

ORCA maximizes the number of feasible velocities near the current speed vectors, whereas RORCA (Wang et al (2021)) maximizes the number of feasible velocities near the zero speed vector. It can be proved that the resulting speed constraints always contains the zero speed vector, ensuring a non-empty solution set for robots that can stop. Despite being useful for ground vehicles and multi-rotor UAVs, it does not provide useful guarantees for commercial air traffic or fixed-wing UAVs.

Durand (2018) highlighted an issue with the original ORCA algorithm when applied to commercial air traffic or fixed-wing UAVs flying at similar constant or nearly-constant speeds: aircraft could end-up flying parallel tracks without ever crossing paths to reach their respective destinations. In the same publication, they proposed a Constant-Speed ORCA algorithm (CS-ORCA) that can better handle these situations.

In this paper, we will discuss the limitations of CS-ORCA, then propose a new logic with two time horizons DH-ORCA that improves on ORCA and CS-ORCA.

3 Context and objectives

3.1 Potential context of application

Geometric algorithms such as ORCA, CS-ORCA, or the DH-ORCA algorithm introduced in this paper can be used in a variety of contexts, with different system architectures.

In a centralized architecture, the geometric algorithm could be implemented in a central computer with the positions and velocities of all traffic as input. The resulting modified velocity vectors

would then be transmitted via data-link to the aircraft or UAVs. Using geometric algorithms in a centralized approach would be fast in terms of computation time, but less optimal in terms of overall trajectory deviations than global optimization methods such as the ones presented in the literature review.

Geometric algorithms are actually intended for a decentralized, autonomous approach. They could be of use either in the context of autonomous airborne separation assurance systems for commercial aircraft or in the *detect & avoid* logic that could be deployed in future Unmanned Airborne Systems. In such a context, each aircraft or UAV computes its own maneuvers on-board, and no explicit coordination among the agents is required. If the same ORCA logic is implemented aboard all agents, and if they all have the same view of the positions and velocities of the traffic, the velocity changes computed aboard each UAV or aircraft are implicitly coordinated (following the implemented logic) with the maneuvers of the other traffic. In other words, the resulting maneuvers are coordinated among all agents without passing or broadcasting any message.

This implies that all agents should apply the same logic, with the same look-ahead time (or time horizon) for the conflict detection. The time horizon parameter(s)[2] should be pre-set before deployment. In section 7.5, we propose a method to tune these parameters, by choosing the values providing the best results on a set of random scenarios.

3.2 Handling the altitude

The original ORCA algorithm from van den Berg et al (2011) was initially designed for a 2D-environment, to simulate robot collision avoidance. As will be detailed in section 4, it works by moving the 2D-vector of relative velocity outside a forbidden domain consisting of a blunt 2D-cone (see Fig. 1).

Aircraft and UAVs fly in a 3D-environment, and several ways have been proposed to adapt the 2D-ORCA algorithm to the 3D-space. The first one in Snape and Manocha (2010) considers a 3D-vector for the relative velocity and a blunt 3D-cone for the velocity obstacle – the head of the 3D-cone being a sphere around the other aircraft.

The second one in Alligier et al (2018) uses vertical cylinders instead of spheres around the aircraft. Each cylinder has a circular basis of radius, the horizontal separation standard, and its height is equal to the vertical separation standard. This model is more consistent with the current operational notion of separation loss in aviation: two flights are not separated when their horizontal distance is closer than the horizontal separation standard, and when their vertical separation is less than the vertical separation standard.

In this paper, we will focus on 2D-separation in the horizontal plane exclusively, and introduce a dual-horizon algorithm that improves on ORCA and CS-ORCA when applied to constant-speed aircraft flying in a horizontal plane. The extension of this dual-horizon algorithm to 3D-space could be made as in Snape and Manocha (2010) or Alligier et al (2018), without difficulty.

3.3 Objective of this paper

The limitations of ORCA in the context of airborne separation come from the fact that the speed range and acceleration of commercial aircraft are small. Whereas robots can reduce their velocity at will and can even stop completely, fixed-winged aircraft and UAVs cannot reduce their speed below the stall speed otherwise they fall to the ground, and their upper speed is limited by their airframe structure and motorization. Actually, fuel-efficiency dictates to operate within much tighter bounds than that. In the ERASMUS speed control project (Bonini et al (2009)), the speed modification range for the cruising speed was set to $[-6\%; +3\%]$ around the nominal speed.

When applying the original ORCA algorithm with a limited speed range, conflicting aircraft flying on converging tracks at similar speeds may end up on parallel tracks, postponing the crossing of their paths for a very long time and potentially never reaching their destination.

The objective of the research presented in this paper is to propose a new algorithm for self-separation in the horizontal plane that is less subject to the parallel-track issue than ORCA or its constant-speed variant CS-ORCA and that helps converging flights to cross paths.

[2]There is only one time horizon for ORCA and CS-ORCA, but the new logic we propose DH-ORCA uses two different time horizons.

As the final intended application is a decentralized autonomous system where each aircraft computes its maneuver without explicit coordination with the others, the algorithm must use only the information available to the own aircraft: typically, the direction toward its own destination and the positions and velocities of the other aircraft obtained through the on-board radar. In particular, we assume that the intended destination of the other aircraft is not known by the own aircraft.

In our traffic scenarios, we will focus on pathological cases where all aircraft or UAVs fly at constant speeds that might differ from one another, but only in a limited range. These scenarios are more prone to exhibit the parallel tracks' behaviour. We here describe these cases as pathological, in the sense that the original ORCA algorithm was not intended to handle them, but such situations may actually occur very often in real life, typically for commercial aircraft flying at similar speeds in the upper airspace. This is not necessarily the case for UAVs flying in lower airspace, as there are a variety of UAVs (fixed-wing, rotor, etc) having different speed and acceleration characteristics. Nonetheless, the self-separation logic of these UAVs should be able to handle conflicts involving UAVs with similar speeds, for safety and efficiency reasons, as such situations may actually occur in the dense traffic situations envisioned in the near future.

4 Optimal Reciprocal Collision Avoidance (ORCA)

This section describes the Optimal Reciprocal Collision Avoidance (ORCA) algorithm developed in van den Berg et al (2011) in the specific case where speeds are constrained. First, the maneuver model is detailed for an aircraft pair, then, the maneuver calculation is explained when more than two aircraft are involved in a conflict.

As explained in section 3, we focus on self-separation in the horizontal plane for aircraft flying at a same altitude, so ORCA and its variants are applied in 2D and provide lateral maneuvers only, in the form of velocity changes. We will assume in this paper that only the direction of the velocity can be changed: the velocity domain is a circular arc (see Fig. 2). All the algorithms presented hereafter can easily be extended to non-constant speeds by considering a velocity domain that is a portion of a disc limited by the minimum and maximum speed and by the turning rate.

4.1 Separation Constraint Model for Two Aircraft

Let us consider two aircraft A and B, as illustrated in Figure 1.a. Let d be the required minimum separation and τ be a time horizon (also called look-ahead time, or anticipation in the literature). Let us denote A and B the respective positions of aircraft A and B at the current time (here $t = 0$ by convention), and $\overrightarrow{v_A}$ and $\overrightarrow{v_B}$ their respective velocity vectors.

Assuming both flights continue on their current course, the future position of aircraft A relative to aircraft B at a future time $t \geq 0$ is given by the vector $\overrightarrow{BA(t)} = \overrightarrow{BA} + (\overrightarrow{v_A} - \overrightarrow{v_B})\,t = \overrightarrow{BA} + \overrightarrow{v_r}\,t$, where $\overrightarrow{v_r}$ is the velocity of A relative to B (*i.e.* considering B fixed).

Given a time horizon τ, and assuming $t = 0$ at the current aircraft positions, there is a conflict between aircraft A and B if and only if we anticipate a separation loss in the future, within the time horizon τ: $\exists t\,,\ 0 \leq t \leq \tau \mid \left\|\overrightarrow{BA(t)}\right\| < d$

If we mentally try to picture, in Figure 1.b, where the extremity of vector $\overrightarrow{BA} + \overrightarrow{v_r}\,t$ lies, we see that it is on a line $\mathcal{L}(A, \overrightarrow{v_r})$ passing through A and directed by $\overrightarrow{v_r}$. If this line $\mathcal{L}(A, \overrightarrow{v_r})$ intersects the circle $\mathcal{C}(B, d)$ of radius d centered on B, then this intersection defines two times t_1 and t_2 at which a future separation loss begins and end: $\forall t \in [t_1, t_2]\ \left\|\overrightarrow{BA(t)}\right\| < d$. This can only happen if $\overrightarrow{v_r}$ lies inside the cone which extremity is A and which sides are the tangents to the circle $\mathcal{C}(B, d)$ passing through A (in grey on the figure). In the case where $\overrightarrow{v_r}$ lies outside this cone of extremity A – which is not the case in Figure 1.b – there will never be a separation loss. To summarize, the fact that $\overrightarrow{v_r}$ lies inside or outside the full cone of extremity A allows us to detect if there is a conflict within an infinite time horizon $\tau = +\infty$, or not.

In order to detect conflicts occurring within a finite time horizon τ, let us now consider in Figure 1.b the smaller circle $\mathcal{C}(B', \frac{d}{\tau})$ of radius $\frac{d}{\tau}$ centered at B' such that $\overrightarrow{AB'} = \frac{\overrightarrow{AB}}{\tau}$. This small circle and the tangents to the big circle

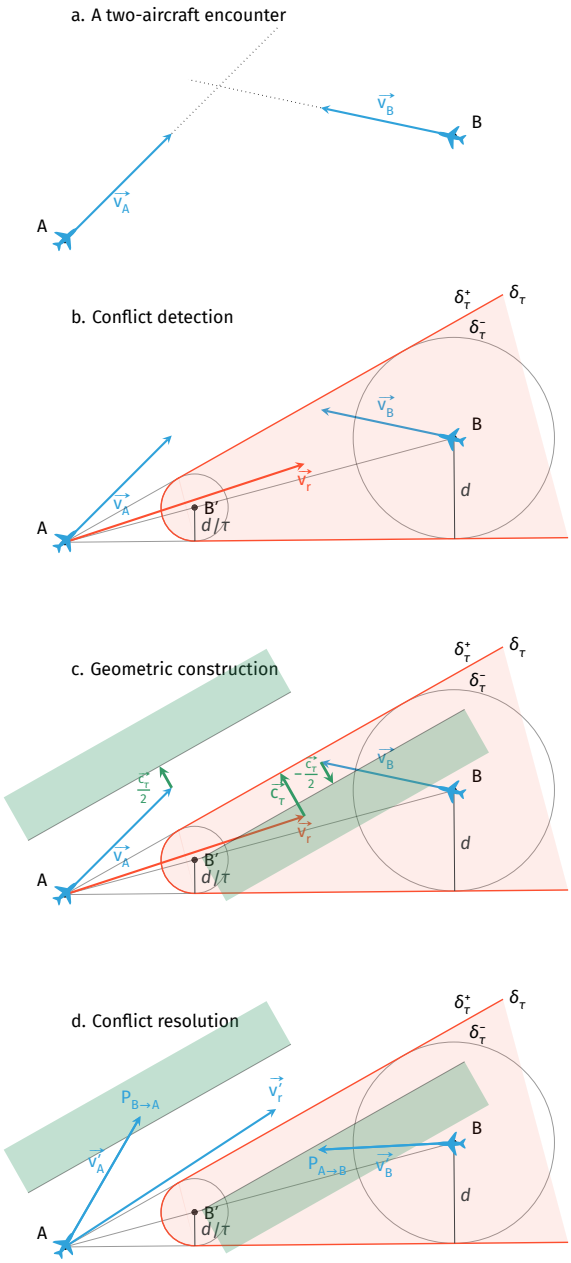


Figure 1: Conflicting aircraft model: a loss of separation will occur within time τ if and only if the relative velocity $\vec{v_r}$ lies in the red forbidden zone.

$\mathcal{C}(B,d)$ define a zone in light red and denoted δ_τ^- in Figure 1.b, bounded by the boundary δ_τ (the bold red curve on the figure). This red zone has the form of a blunt cone (with the small circle at the head) and is called a *reciprocal velocity obstacle*. The relative velocity $\vec{v_r}$ intersects with this zone δ_τ^- if and only if it intersects the disc of size $\frac{d}{\tau}$ centered in B', i.e. if and only if there exists $\lambda \in [0;1]$ such that $\left\|\overrightarrow{B'A} + \lambda\vec{v_r}\right\| < \frac{d}{\tau}$ or $\left\|\overrightarrow{BA} + \lambda\tau\vec{v_r}\right\| < d$ which means that A and B are in conflict at time $\lambda\tau \leq \tau$. Consequently, a conflict will occur within time τ if and only if $\vec{v_r}$ lies in this zone δ_τ^-.

Note that the reciprocal velocity obstacle δ_τ^- lies in the space of relative velocities. When a conflict occurs, it can be avoided simply by modifying the velocities $\vec{v_A}$ and $\vec{v_B}$ such that the relative velocity is moved outside the reciprocal velocity obstacle. The ORCA (Optimal Reciprocal Collision Avoidance) algorithm is based on the principle that the effort to keep $\vec{v_r}$ outside the reciprocal velocity obstacle should be minimal and shared by the two agents A and B. In its original version, the necessary minimal relative velocity change, here denoted $\vec{c_\tau}$, is split equally between the two agents, as shown in Figure 1.c. The vector $\vec{c_\tau}$ is simply obtained by connecting the tip of $\vec{v_r}$ with its orthogonal projection on the closest boundary of the reciprocal velocity obstacle δ_τ. The half-vector $\frac{\vec{c_\tau}}{2}$ is added to $\vec{v_A}$ in order to define a semi-plane $P^\tau_{B\to A}$ perpendicular to $\vec{c_\tau}$, and this same half-vector is subtracted from $\vec{v_B}$ to define a semi-plane $P^\tau_{A\to B}$. Both semi-planes are shown in light green on Figure 1.c.

If the new velocities $\vec{v'_A}$ and $\vec{v'_B}$ are chosen in the semi-planes $P^\tau_{B\to A}$ and $P^\tau_{A\to B}$ respectively, as illustrated in Figure 1.d, the new relative velocity $\vec{v'_r}$ falls outside δ_τ^- and the conflict is solved, at least for a time horizon τ and assuming both aircraft stay on their new course.

The two semi-planes $P^\tau_{B\to A}$ and $P^\tau_{A\to B}$ constraining the velocities of A and B respectively can be formally defined as follows:

$$P^\tau_{B\to A} := \left\{\vec{v'_A} \in \mathbb{R}^2 \mid \left(\vec{v'_A} - \left(\vec{v_A} + \frac{\vec{c_\tau}}{2}\right)\right).\vec{\eta_\tau} \geq 0\right\}$$

$$P^\tau_{A\to B} := \left\{\vec{v'_B} \in \mathbb{R}^2 \mid \left(\vec{v'_B} - \left(\vec{v_B} - \frac{\vec{c_\tau}}{2}\right)\right).\vec{\eta_\tau} \leq 0\right\}$$

In the above definitions, $\vec{\eta_\tau}$ denotes a unit vector perpendicular to the curve δ_τ at the point $\vec{v_r} + \vec{c_\tau}$, and pointing towards the outside of δ_τ^-. Using these equations, we can prove that:

$$\forall \overrightarrow{v'_A} \in P^{\tau}_{B\to A},\ \forall \overrightarrow{v'_B} \in P^{\tau}_{A\to B},$$
$$\left(\overrightarrow{v'_A} - \overrightarrow{v'_B} - (\overrightarrow{v_r} + \overrightarrow{c_\tau})\right).\overrightarrow{\eta_\tau} \geq 0$$

This equation guarantees that $\overrightarrow{v'_r} = \overrightarrow{v'_A} - \overrightarrow{v'_B}$ is outside the red zone δ^-_τ (Figure 1.d). Note that the constraints $P^{\tau}_{B\to A}$ and $P^{\tau}_{A\to B}$ are always enforced, whether a conflict is detected or not between A and B. When no conflict is detected at the current time step, these constraints guarantee that the next velocities chosen for A and B remain conflict-free.

In our description of ORCA, we have chosen to consider the reciprocal velocity obstacle for the relative velocity $\overrightarrow{v_r} = \overrightarrow{v_A} - \overrightarrow{v_B}$, considering B fixed, but we could have reversed the roles of A and B and considered the velocity obstacle for $\overrightarrow{v_B} - \overrightarrow{v_A}$, considering A fixed. This would have given exactly the same semi-planes and the same result. This computation can be made on a central computer deciding the maneuvers for both aircraft, but more interestingly it can also be made separately onboard the two aircraft, with each aircraft deciding of its own maneuver within its conflict-free semi-plane. If both aircraft share the same view of the situation and apply the same logic, their maneuvers will be implicitly coordinated, without having to exchange messages.

4.2 Speed and Turn Rate Constraints

Track changes are the only possible maneuvers for the aircraft or fixed-wing UAVs considered in this study. The aircraft speed remains constant throughout the conflict resolution process. This can be expressed as the following constraints on the velocities of aircraft A and B:

$$\left\|\overrightarrow{v'_A}\right\| = \|\overrightarrow{v_A}\|$$

$$\left\|\overrightarrow{v'_B}\right\| = \|\overrightarrow{v_B}\|$$

In addition, the turn rate of every aircraft is limited to a maximum value, which can be specific to each aircraft. In the current study, it was limited to 3 degrees per second for all aircraft.

With the above constraints, the new velocity assigned by ORCA to an aircraft must be chosen so that the head of the velocity vector is on an arc of limited range, centered on the current aircraft position (see Figure 2). The arc range is limited by the maximum turning rate. If the time step is set to 5 seconds, for example, the arc range is ±15 degrees for the standard turning rate (3 degrees per second).

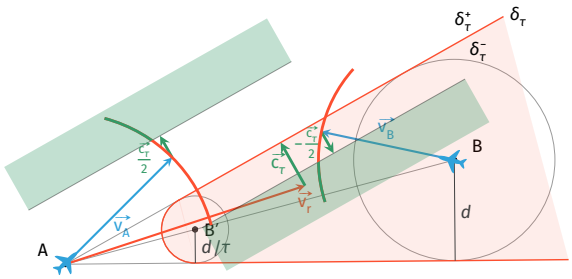


Figure 2: Conflicting aircraft model: the effort is shared by the two aircraft. The new velocities must be chosen on the green portion of the arc.

Considering the separation constraints $P^{\tau}_{B\to A}$ and $P^{\tau}_{A\to B}$ presented in the previous section 4.1, we see in Figure 2 that the velocity vectors must be chosen in the green portions of the arcs, inside these semi-planes, in order to move the relative velocity vector outside the red blunt cone δ^-_τ while satisfying the speed constraints.

4.3 Multiple-Aircraft Separation Constraints

When an aircraft i is in conflict with more than two aircraft, the intersection of the semi-planar separation constraints $P^{\tau}_{k\to i}$ with $k \neq i$ creates a 2d polyhedral set C_i. This 2d polyhedral set C_i might be unbounded. This is the case for example in Figure 3 where A is conflicting with B and D. The conflict-free convex set C_A for aircraft A is the intersection of the two semi-planes $P^{\tau}_{B\to A}$ and $P^{\tau}_{D\to A}$.

The intersection of C_i and the arc A_i of possible turning angles for aircraft i (Section 4.2) defines an arc S_i of admissible conflict-free velocities for aircraft i. If every aircraft selects its new velocity in this conflict-free arc, then no loss of separation will occur in the next τ seconds.

If, for a given aircraft i, the intersection of the semi-planes is empty, then every semi-plane is equally slightly moved until a non-empty intersection appears. The resulting convex C_i does not guarantee a conflict-free maneuver anymore, but it remains close to the conflict-free domain. This

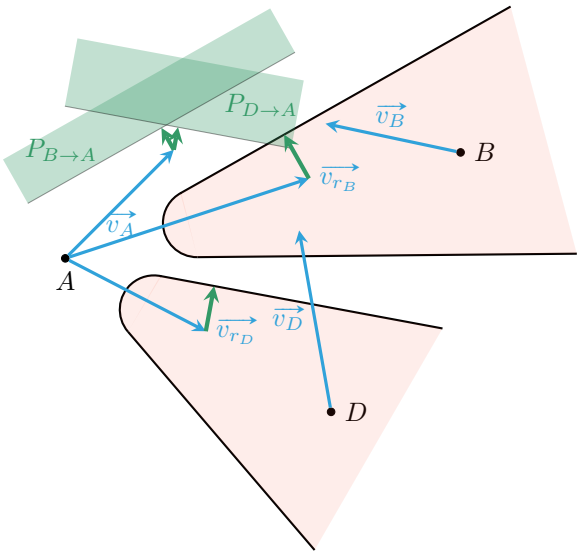


Figure 3: Multi-Conflicting aircraft model: for aircraft A, a conflict will occur within time τ with aircraft B (resp. D) if and only if the relative velocity $\overrightarrow{v_{r_B}}$ (resp. $\overrightarrow{v_{r_D}}$) lies in the forbidden zone in red associated with aircraft B (resp. D).

process is described by Van den Berg van den Berg et al (2011).

4.4 Summary of the ORCA algorithm applied to constant-speed aircraft

Algorithm 1 summarizes the ORCA algorithm applied to constant-speed aircraft (or UAVs). Time is discretized using a time step δt (see Table 1 in section 7.3 for the parameter values chosen in the experiments). As long as every aircraft has not reached its destination, every aircraft pair (i, j) is checked to calculate the semi-planes $P_{j\to i} = P^{\tau}_{j\to i}$ and $P_{i\to j} = P^{\tau}_{i\to j}$. For every aircraft i, the convex C_i is computed as the intersection of the semi-planes $P_{k\to i}$, with $k \neq i$. The admissible domain S_i for the new velocity is then the intersection of C_i and the arc A_i of possible turning angles for aircraft i.

The new velocity $\overrightarrow{v_i'}$ is chosen as the closest vector to $\overrightarrow{v_i}^{\text{pref}}$ in the admissible domain S_i, where $\overrightarrow{v_i}^{\text{pref}}$ is the preferred velocity vector for aircraft i, directed from the aircraft current position towards its destination.

Note that the admissible velocity domain S_i can be empty for two reasons: either the convex C_i is not empty but its intersection with A_i is empty, or the convex C_i is empty. In both cases, we select the new velocity $\overrightarrow{v_i'}$ that violates the constraints as little as possible. This is done by incrementally relaxing the constraints – as explained at the end of the previous subsection 4.3 – until S_i is not empty.

Algorithm 1 ORCA algorithm, applied to constant-speed aircraft

Input: simulation time step δt, time horizon τ, and a scenario with the planned origin, destination, cruising speed and scheduled entry time of each aircraft

Output: aircraft trajectories

while every aircraft has not reached destination **do**
  for every aircraft couple (i, j) **do**
    Define the semi-plane constraints $P_{j\to i}$ and $P_{i\to j}$
  end for
  for every aircraft i **do**
    Calculate the convex intersection C_i of every semi-planes $C_i = \bigcap_{k\neq i} P_{k\to i}$
    Calculate the arc A_i of possible aircraft i turning range.
    Calculate $S_i = C_i \bigcap A_i$
    Calculate the ideal velocity $\overrightarrow{v_i}^{\text{pref}}$
    Choose $\overrightarrow{v_i'}$ the velocity closest to $\overrightarrow{v_i}^{\text{pref}}$ inside S_i
    Change current velocity: $\overrightarrow{v_i} = \overrightarrow{v_i'}$
    Move aircraft i with current velocity $\overrightarrow{v_i}$
  end for
  increase current time $t = t + \delta t$
end while

5 Constant-Speed ORCA (CS-ORCA)

This section describes the Constant-Speed ORCA algorithm (CS-ORCA) introduced by Durand in Durand (2018). This variant aims at correcting the pathological behaviour exhibited by the original ORCA algorithm when constant-speed aircraft converge at a small angle Durand and Barnier (2015). In such cases, ORCA tends to postpone the conflict outside the time horizon τ, indefinitely, by setting the flights on parallel tracks.

We will first explain this behaviour in more detail in Section 5.1, then give a brief description of CS-ORCA (Section 5.2), before explaining in Section 5.3 why CS-ORCA does not completely solve the pathological cases it was designed to solve.

5.1 Pathological solutions where flights cannot cross paths

In Figure 1, we presented a conflict situation where $\vec{v_r}$ lied deep within the forbidden zone δ_τ^- (the blunt cone in red), and the vector $\vec{c_\tau}$ computed by ORCA to move $\vec{v_r}$ outside δ_τ^- was perpendicular to one of the cone sides.

Let us now consider Figure 4 showing a conflict in which the relative velocity $\vec{v_r}$ is close to the circular part of the blunt cone δ_τ^- (in red). Such situations are more likely to occur at the beginning of a conflict.

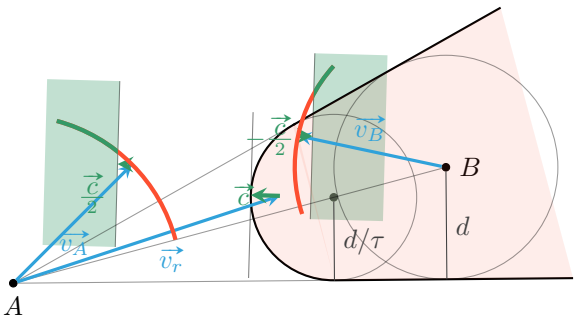


Figure 4: Example where the tip of the relative velocity $\vec{v_r}$ is projected onto the circular part of the reciprocal velocity obstacle (in red), leading to ORCA velocity constraints (semi-planes in green) that move the velocities of A and B apart, making them more parallel.

In such cases, the original ORCA algorithm computes a vector $\vec{c_\tau}$ joining the tip of $\vec{v_r}$ to the closest point on the circular part of the frontier δ_τ of the velocity obstacle. The direction of $\vec{c_\tau}$ is not perpendicular to either one of the cone sides, and the semi-planes defining the constraints for $\vec{v_A}$ and $\vec{v_B}$ – which are perpendicular to $\vec{c_\tau}$ – are not parallel to either one of the cone sides. They actually tend to move the velocities of A and B slightly apart.

This move ensures that there will be no loss of separation within the next τ seconds. However, the conflict may reappear in the next time steps—and be solved in the same way—if the aircraft are still converging. This may typically occur when the aircraft must cross paths to reach their respective destination.

As a result, depending on the initial conditions, the two aircraft velocities tend to become parallel. Instead of solving the conflict by crossing the paths of the two aircraft, thus allowing them to reach their destination, the ORCA algorithm tends to postpone indefinitely the conflict outside the time horizon by setting them on parallel tracks. This is called the horizon effect in game theory.

5.2 The CS-ORCA algorithm

To deal with the problem described in the previous section, the semi-planes in CS-ORCA are computed differently than with ORCA, but the principle of the algorithm is the same. The finite time horizon τ is used to detect conflicts (i.e. future separation losses), exactly as in ORCA, by checking if the relative velocity $\vec{v_r}$ lies inside the reciprocal velocity obstacle (the blunt red cone, on Fig.4, with a circle of radius $\frac{d}{\tau}$ at the head, and denoted δ_τ^- in Section 4.1).

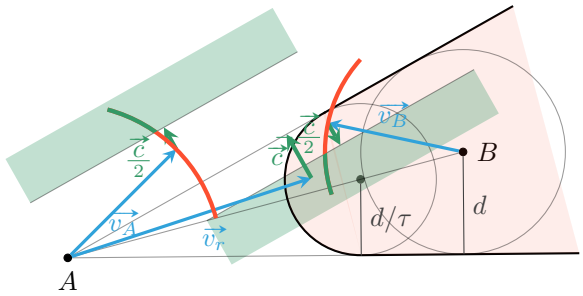


Figure 5: CS-ORCA velocity constraint model: the relative speed $\vec{v_r}$ is now projected on the closest side of the cone instead of the circular part, only when $\vec{v_r}$ is inside the forbidden zone (in red).

When no conflict is detected, *i.e.* when $\vec{v_r}$ lies outside δ_τ^-, the semi-planes $P^\tau_{A\to B}$ and $P^\tau_{B\to A}$ are defined as in ORCA, by considering the orthogonal projection of the tip of $\vec{v_r}$ onto the closest boundary of δ_τ^-. In this case, these semi-planes

prevent from selecting new velocities that would lead to a conflict at the next time step. Depending on how $\overrightarrow{v_r}$ is positioned outside δ_τ^-, the semi-planes $P^\tau_{A\to B}$ and $P^\tau_{B\to A}$ might be parallel to one of the straight sides of δ_τ^-, or parallel to a tangent to the round head of δ_τ^-.

The difference between ORCA and CS-ORCA lies in how the semi-planes implementing the velocity constraints are computed when a conflict is detected, *i.e.* when $\overrightarrow{v_r}$ lies inside δ_τ^-. This difference is illustrated in Figures 4 and 5. Figure 4 shows how ORCA builds the semi-planes by projecting the tip of $\overrightarrow{v_r}$ onto the closest boundary of δ_τ^- – which can be the round part at the head of the cone δ_τ^-, as in Fig. 4. Figure 5 shows how CS-ORCA builds the relative velocity deviation $\overrightarrow{c}$ by projecting the tip of $\overrightarrow{v_r}$ onto the closest side of the full cone of extremity A, instead of the closest boundary of the blunt cone. Remembering from Section 4.1 that this full cone is in fact the reciprocal velocity obstacle corresponding to an infinite time horizon $\tau = +\infty$[3], we can denote $P^{\tau=+\infty}_{B\to A}$ and $P^{\tau=+\infty}_{A\to B}$ the semi-planes defined using this full cone $\delta^-_{\tau=+\infty}$.

With these notations, the semi-plane implementing the constraint induced by aircraft B on the velocity of aircraft A can be mathematically summarized as follows for CS-ORCA:

$$P^{\text{CSORCA}_\tau}_{B\to A} := \begin{cases} P^{\tau=+\infty}_{B\to A} & \text{if } \overrightarrow{v_r} \in \delta^-_\tau \\ P^{\tau=\tau}_{B\to A} & \text{otherwise} \end{cases}$$

CS-ORCA is just Algorithm 1 with $P_{i\to j} = P^{\text{CSORCA}_\tau}_{i\to j}$ for all aircraft pairs (i,j) with $i \neq j$.

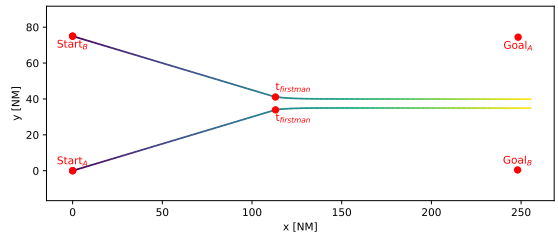
As shown in Figure 5, with CS-ORCA the two aircraft turn left, as opposed to the choice made with ORCA in Figure 4. This allows both aircraft to reach their destination, without postponing the conflict beyond the time horizon.

5.3 Issue with CS-ORCA

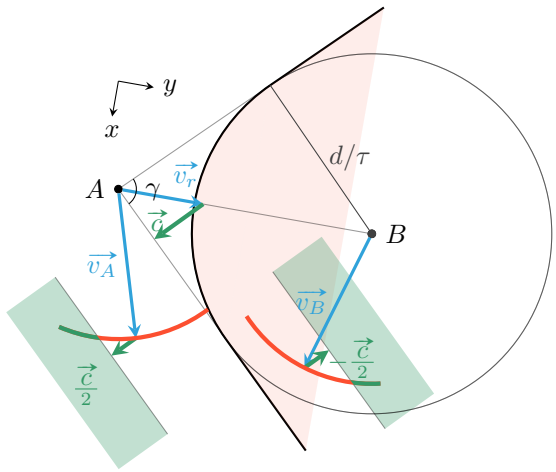
Let us now explain why CS-ORCA does not always solve the pathological situations, exposed in section 5.1, that it was intended to solve. When aircraft have to cross one another, τ must be chosen large enough in order to avoid the parallel behaviour. To understand this, we can look at a situation with only two aircraft with the exact same speed converging at a small angle, as illustrated in Figure 6.

If τ is too small, CS-ORCA chooses parallel tracks, the conflict is avoided, but aircraft do not

[3]One can also realize that by considering how the shape of δ_τ^- is modified when increasing τ to $+\infty$: B' moves to the position of A, and the radius of the round head of δ_τ^- tends to 0.

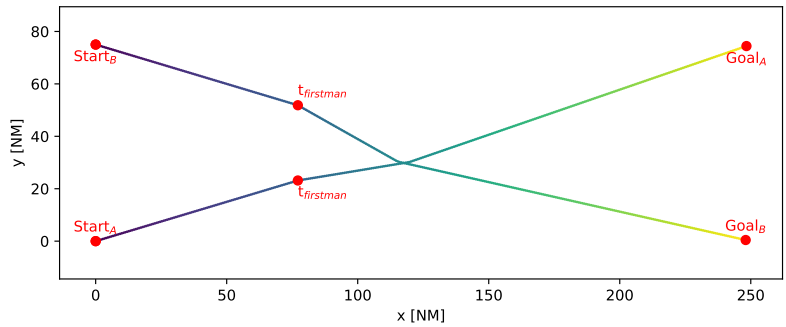


(a) With $\tau = 1$ min, CS-ORCA sets the two aircraft on parallel courses. The conflict is avoided, but the aircraft do not reach their destinations. Figure 6b depicts the first maneuver at $t_{firstman}$ that led to this situation.

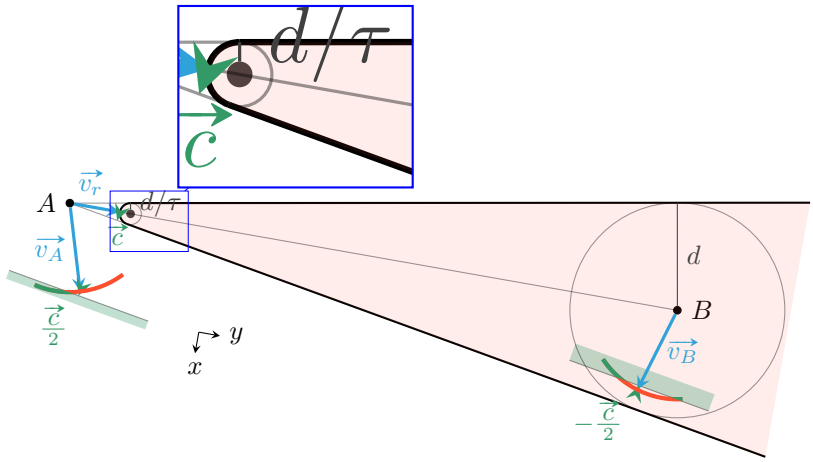


(b) First maneuver performed at $t_{firstman}$, when $\overrightarrow{v_r}$ first enters the forbidden cone. Here, this happens when the aircraft are close. As a consequence, the cone angle is large. The resulting constraints (green semi-planes and arcs) tend to move the velocities toward parallel directions. The x and y directions are shown so that it is easier to relate this figure with Fig. 6a.

Figure 6: Example with $\tau = 1$ min where CS-ORCA sets two aircraft on parallel tracks (Fig. 6a), by moving the velocities apart (Fig. 6b). Figure 6b is scaled to improve readability by choosing the minute as unit of time. This explains why B is here the center of both the circles of radius d and of radius $\frac{d}{\tau}$.



(a) With $\tau = 10$ min, CS-ORCA makes both aircraft turn right, the conflict is avoided, and the aircraft reach their destinations. Figure 7b depicts the situation at $t_{firstman}$ that explains this choice.



(b) First maneuver performed at $t_{firstman}$. Here, the aircraft are far from each other when $\overrightarrow{v_r}$ first enters the forbidden cone. As a consequence, the cone angle is small. The resulting constraints (semi-planes and arcs in green) make both aircraft turn right, here increasing their relative closing speed, which helps them to cross paths. The x and y directions are shown so that it is easier to relate this figure with Figure 7a.

Figure 7: Example with $\tau = 10$ min where CS-ORCA makes both aircraft turn right and reduce their relative closing speed, allowing them to cross paths and reach their destination.

reach their destinations as depicted in Figure 6a. As can be seen in Figure 6b, the first maneuver begins when $\overrightarrow{v_r}$ enters the forbidden zone δ_τ^-. With a small τ, aircraft are close when this happens.

As a consequence, the angle $\gamma = 2 \arcsin \frac{AB}{d}$ made by the cone is large. Thus, the angle between the chosen semi-planes and $\overrightarrow{AB}$, equal to $\frac{\gamma}{2}$, is large too. This results in constraints forcing a reduction of the velocity vectors' component along the vector $\overrightarrow{AB}$: the aircraft reduce their closing speed by making their velocities parallel, as plotted in Figure 6b.

Conversely, if τ is large enough, CS-ORCA makes them both turn right (or left), and both aircraft reach their destination as illustrated in Figure 7a showing the same example as in Fig. 6b, with a larger time horizon $\tau = 10$ min. The maneuver begins when $\overrightarrow{v_r}$ enters the forbidden cone as depicted in Figure 7b. With a large τ, the aircraft are far from each other when this happens.

As a consequence, the angle between the chosen semi-planes and $\overrightarrow{AB}$ is small, thus favouring maneuvers with a non-null closing speed and allowing aircraft to cross paths.

Ideally, one would want a large τ when aircraft have to cross one another, in order to avoid the parallel behaviour. When aircraft do not want to cross one another, a smaller τ would result in less restrictive constraints and be sufficient to avoid conflict efficiently.

6 Dual-Horizon ORCA (DH-ORCA)

We have seen in the previous section that CS-ORCA was initially meant to avoid the parallel behaviour when aircraft have to cross one another, but in order to work a large enough value of τ must be chosen. The drawback of having a large value of τ is that it will be used in all situations, even ones that do not require aircraft to cross paths.

With a large τ, we have a loss of efficiency – as the constraints imposed on the velocity vectors are more restrictive than required – and with a small τ, the CS-ORCA algorithm exhibits the pathological behaviour we are trying to avoid.

With this in mind, we propose a new logic, called Dual-Horizon ORCA (DH-ORCA), with two time horizons. A small value τ_{conflict} is used to enforce systematically the standard ORCA constraints. A larger value $\tau_{\text{cross}} > \tau_{\text{conflict}}$ is used to enforce optional CS-ORCA-like constraints.

The optional constraint is enforced by an aircraft only when its preferred course towards its destination crosses the current trajectory of another aircraft, and is in conflict with it in the τ_{cross} time horizon. Note that this logic does not require the aircraft to broadcast their preferred course to others: Each aircraft only needs its own intent to decide whether to enforce the optional constraint.

6.1 The DH-ORCA Constraints and Algorithm

The two constraints created by τ_{conflict}, one for each aircraft, are always enforced. This guarantees that there will be no loss of separation within a time horizon τ_{conflict}. These constraints are denoted $P_{\rightarrow}^{\tau=\tau_{\text{conflict}}}$. They are less restrictive than the $P_{\rightarrow}^{\tau=+\infty}$ constraints of CS-ORCA.

The two constraints created by τ_{cross}, one for each aircraft, are here to avoid the parallel behaviour and will be created using the cone side closest to $\overrightarrow{v_r}$, as in CS-ORCA. Let us denote $P_{\rightarrow}^{\tau=+\infty}$ these crossing constraints. Because the separation constraints $P_{\rightarrow}^{\tau=\tau_{\text{conflict}}}$ created with the shortest time horizon τ_{conflict} are sufficient to guarantee conflict-free trajectories—at least in the τ_{conflict} time horizon—, the constraints created by τ_{cross} are somewhat optional. In fact, each aircraft decides if it enforces or ignores the crossing constraint generated by the other.

Making the $P_{\rightarrow}^{\tau=+\infty}$ constraint optional avoids to restrict the velocity choices when not necessary. Here, the constraint will be enforced by an aircraft only when its preferred course towards its destination crosses the current trajectory of the other aircraft and if a loss of separation is predicted to occur in the τ_{cross} time horizon when following this preferred course.

Let us denote $\text{Cross}(A, B)$ this condition, allowing A to decide if it should enforce a $P_{B\rightarrow A}^{\tau=+\infty}$ constraint because of aircraft B. This condition is expressed in the following equation, where d is the required minimum separation and $\overrightarrow{v_A}^{\text{pref}}$ is the preferred velocity vector of aircraft A, directed toward its destination:

$$\text{Cross}(A,B) :=$$
$$\exists\, t \in [0, \tau_{\text{cross}}]\ s.t.\ \left\| \overrightarrow{BA} + (\overrightarrow{v_A}^{\text{pref}} - \overrightarrow{v_B})\, t \right\| < d$$

Note that aircraft A assumes that B keeps following its current velocity vector $\overrightarrow{v_B}$. Using $\overrightarrow{v_B}^{\text{pref}}$ instead of $\overrightarrow{v_B}$ would more accurately model a conflict between the intended paths of both aircraft, but would require aircraft B to share its intended velocity with A. Here $\overrightarrow{v_B}$ is used as a proxy of $\overrightarrow{v_B}^{\text{pref}}$. If $\overrightarrow{v_B}$ is not close to $\overrightarrow{v_B}^{\text{pref}}$ then $\text{Cross}(A, B)$ could be false despite aircraft A and B having to cross one another to go to their destination. Note also that the $\text{Cross}(.,.)$ criteria is not symmetric: $\text{Cross}(A, B)$ might have a value different from $\text{Cross}(B, A)$.

The constraint $P_{B\rightarrow A}^{\tau=+\infty}$ generated by B is enforced by A if and only if $\text{Cross}(A, B)$ is true.

The DH-ORCA constraints can be mathematically summarized as follows:

$$P_{B\to A}^{\mathrm{DH-ORCA}^{\tau_{\mathrm{cross}}}_{\tau_{\mathrm{conflict}}}} :=$$

$$\begin{cases} P_{B\to A}^{\tau=\tau_{\mathrm{conflict}}} \bigcap P_{B\to A}^{\tau=+\infty} & \text{if } \mathrm{Cross}(A,B) \\ P_{B\to A}^{\tau=\tau_{\mathrm{conflict}}} & \text{otherwise} \end{cases}$$

DH-ORCA is just Algorithm 1 with $P_{i\to j} = P_{i\to j}^{\mathrm{DH-ORCA}^{\tau_{\mathrm{cross}}}_{\tau_{\mathrm{conflict}}}}$ for all aircraft pairs (i,j) with $i \neq j$.

6.2 Geometrical Representation of the DH-ORCA constraints

A geometrical representation of the Cross(.,.) criteria is also possible: $\mathrm{Cross}(A,B)$ is true if and only if $\overrightarrow{v_A}^{\mathrm{pref}} - \overrightarrow{v_B}$ is inside the τ_{cross} forbidden zone: $\overrightarrow{v_A}^{\mathrm{pref}} - \overrightarrow{v_B} \in \delta^-_{\tau_{\mathrm{cross}}}$. Using the geometrical interpretation of this criterion, Figure 8 depicts the three possible cases for the $P_{\rightarrow}^{\tau=+\infty}$ constraints:

1. Both aircraft ignore their $P_{\rightarrow}^{\tau=+\infty}$ constraint as depicted in Figure 8a: only the classic constraints from ORCA with τ_{conflict} are considered. This gives more freedom to the aircraft to modify their velocity. The new velocities are chosen as if only ORCA was used with $\tau = \tau_{\mathrm{conflict}}$. This case is typically encountered when the aircraft are far from each other, or when the direction of the preferred velocity of each aircraft diverges from the other aircraft's current velocity.
2. Only one of the two aircraft enforces the crossing constraint $P_{\rightarrow}^{\tau=+\infty}$, as depicted in Figure 8b: this crossing constraint is considered along with the classic ORCA-style separation constraints. See subsection 6.3 for a discussion on the effect of the asymmetrical enforcement of the DH-ORCA constraints.
3. Both $P_{\rightarrow}^{\tau=+\infty}$ constraints are enforced as depicted in Figure 8c: Both aircraft must choose a velocity satisfying $P_{\rightarrow}^{\tau=+\infty}$. As a consequence, the new relative velocity $\overrightarrow{v'_r}$ shall fall outside the cone $\delta^-_{\tau=+\infty}$, allowing the aircraft to safely cross one another.

6.3 Discussion on the Asymmetry of DH-ORCA Constraints

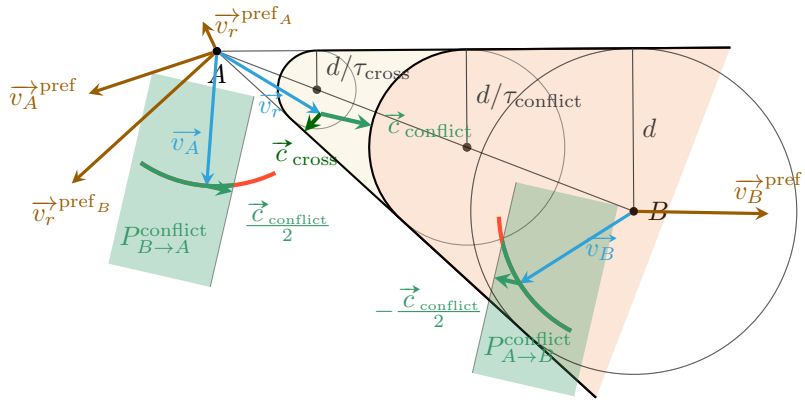
When both aircraft enforce their $P_{\rightarrow}^{\tau=+\infty}$ constraint, as on Figure 8c, both aircraft change their velocity vector to move $\overrightarrow{v'_r}$ outside the $\delta^-_{\tau=+\infty}$ cone. The maneuver is equally split among the two aircraft and each aircraft do its part to avoid the other. If both aircraft do this, $\overrightarrow{v'_r}$ is guaranteed to be outside the cone $\delta^-_{\tau=+\infty}$ and consequently no conflict will ever happen assuming all aircraft stay on their course.

In Figure 8b, where only one aircraft enforces its $P_{\rightarrow}^{\tau=+\infty}$ constraint, we do not have such a guarantee: only one aircraft is actively maneuvering to avoid the other. However, it was experimentally observed that having only one aircraft maneuvering is sometimes enough to avoid the parallel behaviour. Roughly speaking, at each time step, the constrained aircraft takes half of the required speed change. It might not be enough to actually move $\overrightarrow{v_r}$ outside the $\delta^-_{\tau=+\infty}$ cone, but it tries to move $\overrightarrow{v'_r}$ toward the edge of the cone: $\delta_{\tau=+\infty}$. As aircraft are still far from each other, the enforced $P_{\rightarrow}^{\tau=+\infty}$ constraint do not drastically reduce the aircraft's speed component along $\overrightarrow{AB}$, thus avoiding the parallel behaviour. Furthermore, having $\overrightarrow{v_r}$ close to $\delta_{\tau=+\infty}$ somewhat pre-solves the crossing problem: as time goes by, $\overrightarrow{v_r}$ will eventually get inside $\delta^-_{\tau=\tau_{\mathrm{conflict}}}$ but as $\overrightarrow{v_r}$ is close to $\delta_{\tau=+\infty}$, chances are that the minimal maneuvers to get $\overrightarrow{v'_r}$ outside $\delta^-_{\tau=\tau_{\mathrm{conflict}}}$ will be also maneuvers that get $\overrightarrow{v'_r}$ outside $\delta^-_{\tau=+\infty}$, allowing aircraft to cross one another.

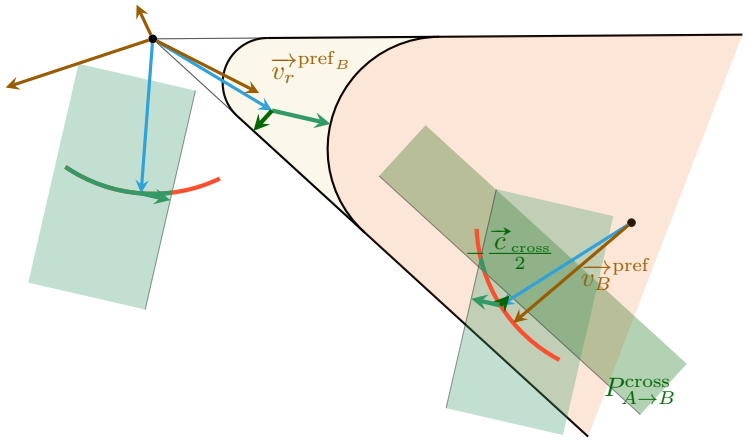
6.4 Illustration of the Behavior of ORCA, CS-ORCA and DH-ORCA on a simple example.

Let us consider the simple example shown in Figure 9a where 5 flights cruising at 230 kts at the same altitude must cross paths to reach their respective destinations. The separation standard d is 5 NM. The departure points (labeled "Start") are evenly spaced apart by a distance of 25 NM, as well as the arrival points (labeled "Goal").

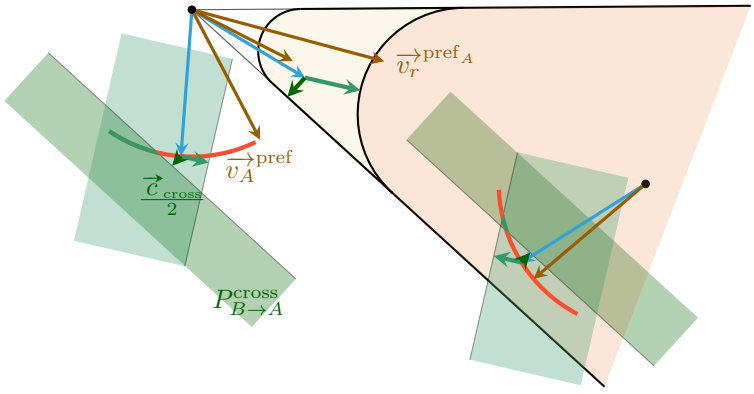
When no *Detect & Avoid* algorithm is applied (Fig. 9a), all 5 aicraft fly directly to their destinations, but separation losses occur, with a total duration of 605 s.



(a) In this example, the $\overrightarrow{v_r}^{\text{pref}}$ are not inside the τ_{cross} forbidden zone, no P^{cross} constraints are added.

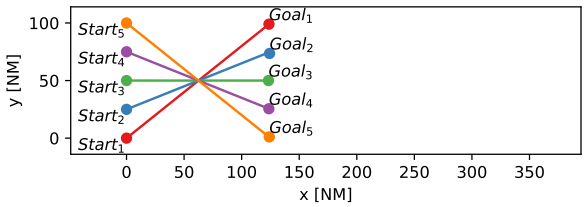


(b) In this example, $\overrightarrow{v_r}^{\text{pref}_B}$ is inside the τ_{cross} forbidden zone, $P^{\tau=+\infty}_{A\to B}$ is added.

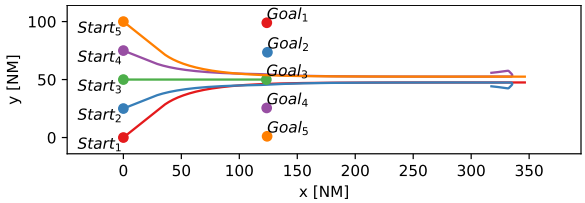


(c) In this example, both $\overrightarrow{v_r}^{\text{pref}_A}$ and $\overrightarrow{v_r}^{\text{pref}_B}$ are inside the τ_{cross} forbidden zone, both $P^{\tau=+\infty}_{A\to B}$ and $P^{\tau=+\infty}_{B\to A}$ are added.

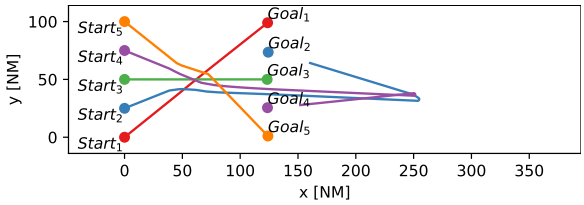
Figure 8: Three examples with different preferred velocities are considered. They illustrate the three possible cases concerning the P^{cross} constraints. Let us note $\overrightarrow{v_r}^{\text{pref}_A} = \overrightarrow{v_A}^{\text{pref}} - \overrightarrow{v_B}$ and $\overrightarrow{v_r}^{\text{pref}_B} = \overrightarrow{v_A} - \overrightarrow{v_B}^{\text{pref}}$. Reading from Figure 8a to Figure 8c, only the modified elements are labelled in order to improve readability.



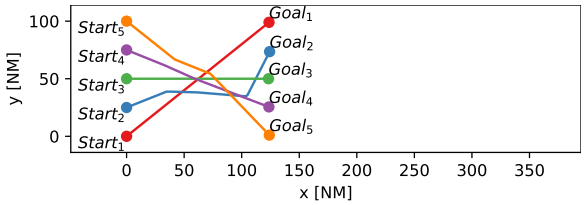
(a) No resolution is performed here, resulting in a 605 s separation loss duration.



(b) ORCA algorithm is used, no loss of separation and two parallel behavior are observed on this simple scenario.



(c) CS-ORCA algorithm is used, no loss of separation and one parallel behavior are observed on this simple scenario.



(d) DH-ORCA algorithm is used, no loss of separation and no parallel behavior are observed on this simple scenario.

Figure 9: An example in which the parallel behavior is observed with ORCA and CS-ORCA but not with DH-ORCA.

When applying a *Detect & Avoid* algorithm (Figures 9b to 9d) with the τ-values tuned in section 7.5 and found in Table 4, no separation loss remains, whatever the chosen algorithm (ORCA, CS-ORCA, or DH-ORCA). However, we see that both algorithms ORCA (Fig. 9b) and CS-ORCA (Fig. 9c) exhibit the pathological parallel-tracks behavior described in section 5.1, whereas DH-ORCA does not (Fig. 9d)

Of course, we cannot draw general conclusions from this simple illustration and DH-ORCA does not guarantee that the pathological solutions will never occur. In the following, we will show experimentally in Section 8 that DH-ORCA drastically reduces the occurrences of pathological solutions when compared with ORCA and CS-ORCA on hours of random scenarios, described in next Section 7, especially for aircraft flying at the same speed.

7 Traffic Scenarios, Simulation, and Experiment Setup

7.1 Traffic Scenarios (Flight Plans)

In our experiments, we compare the different methods described in the previous sections on random traffic scenarios.

We consider a circular area of chosen radius R. A new flight enters the area at a random location on the boundary, taken in the admissible domain determined by exclusion zones around the positions of the preceding flights, so as to avoid conflicting situations on entry. More precisely, as illustrated in Figure 10, we extrapolate the positions of the flights already in the area at the time t when the new flight is scheduled to enter the area, assuming each flight follows its initial straight route toward its destination. We then define a circular exclusion zone of chosen radius (greater than the separation standard d) around each extrapolated position. The admissible entry domain (in green on Figure 10) is obtained by removing the intersections between the outer boundary and the exclusion zones.

Let us denote Δ_{entry} the separation that we wish to enforce between an entering aircraft and the aircraft already in the area. To account for the fact that the traffic is simulated with a user-chosen time step δt, the exclusion zones defined around the positions of these other aircraft must have a radius $\Delta_{entry} + V\,\delta t$, where V is the speed of the entering flight. The additional margin $V\,\delta t$ accounts for the fact that the entering aircraft actually enters the area at a time t_{start} between t

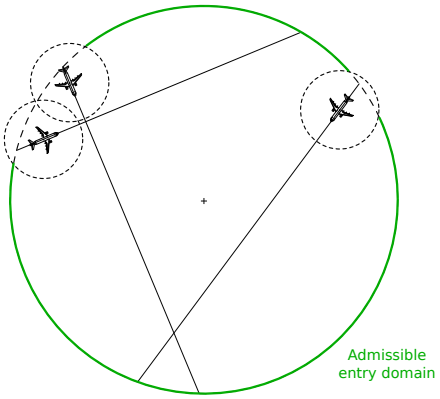


Figure 10: Admissible domain for an entry point (in green), and exclusion zones around flights already in the area.

and $t + \delta t$ – where t is the time of the current simulation step – and it flies over a certain distance between t_{start} and $t + \delta t$, that could put it within a circle of radius Δ_{entry} around another flight already in the area. In our experiments, we chose four times the separation standard d for Δ_{entry}

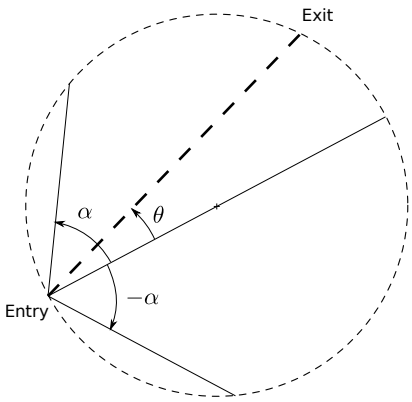


Figure 11: Route selection, at a random angle θ in $\mathcal{U}([-\alpha, \alpha])$ from the direction towards the center of the area. The exit point is determined by the selected route.

Once the entry point is chosen, the flight will follow a straight route at a random angle θ from the direction towards the center of the area (see Fig. 11). This random angle θ is drawn from a uniform distribution $\mathcal{U}([-\alpha, \alpha])$. The destination of each flight is the intersection of this straight route and the circular boundary opposite to the entry point. It can be easily shown that the average crossing distance of the circular area is $\frac{2R\sin\alpha}{\alpha}$ when $\alpha > 0$ and $2R$ when $\alpha = 0$.

The aircraft speed V is randomly chosen between a minimum value V_{min} and a maximum value V_{max}. We assume no wind.

The average crossing time is then $\frac{2R\sin\alpha}{\alpha} / \left(\frac{V_{min} + V_{max}}{2}\right)$.

In all random scenarios, the flight entry times follow a Poisson distribution of parameter λ (the desired average number of flight entries per unit of time). By definition of the Poisson distribution, the average number of flights entering the area over a given period of time T is λT and the variance of this number of entering flights is also λT.

We are interested in measuring some performance metrics at a given average traffic density. The traffic density is the number of aircraft or drones within the area at a given moment. This number depends on the inbound and outbound flows. Assuming the circular area is supplied in flights between t_{start} and t_{end} following the Poisson distribution, there will be a transition period at the beginning of the scenario, when the traffic density increases while the outbound flow remains at zero. After this transition period, the traffic density remains approximately stable – with some variations mainly due to the Poisson distribution of the inbound flow and also to the distribution of the crossing times – before decreasing after t_{end} until it reaches zero at time t_f, the end of the simulation. Figure 12 illustrates the typical evolution of the traffic density when the incoming flow follows a Poisson distribution, over a period of 24 hours.

To summarize, a scenario is characterized by several parameters: radius of circular area considered in the experiment, average number of flights entering the circular area per unit of time, duration of the scenario, upper and lower bounds for the ground speed, angle α defining the angular bounds for the route, radius Δ_{entry} of the exclusion zones for the entry conflicts.

Note that the scenarios are designed so that all flights reach their exit point before the end of the simulation t_f when the flights follow their initial straight route as initially planned (when no D&A maneuver is performed).

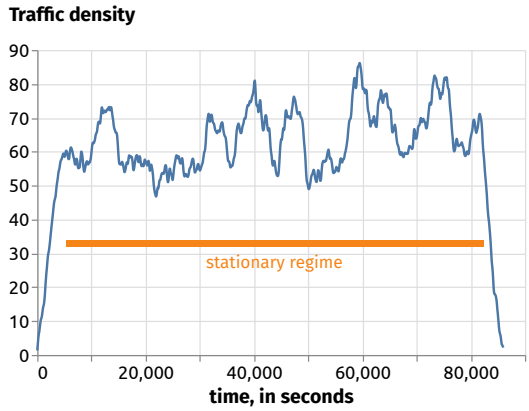


Figure 12: Typical evolution of the traffic density in a random scenario where flight entry times are sampled from a Poisson distribution. The stationary regime is shown by the horizontal orange segment.

7.2 Traffic Simulation with a Detect and Avoid (D&A) module

We have developed a small generic fast-time traffic simulator to execute the traffic scenarios and evaluate several detect and avoid algorithms such as the ones described in previous sections. Any D&A algorithm or conflict resolution algorithm can be easily plugged in the generic simulator to provide maneuvering instructions to the flights every δt seconds, where δt is the user-chosen time step of the simulation.

A D&A algorithm may modify the trajectory of a flight by changing the norm and/or the direction of its velocity at any time step of the simulation. However, in the current experiment, the norm is kept constant, and we only change the direction of the velocity vector. Note that whatever the trajectory modifications, the objective of a flight still remains to reach the exit point that was initially planned, so the D&A algorithm should at some point allow it to resume its navigation towards this exit point, unless it is unable to because of other traffic.

The random scenarios described in section 7.1 are designed such that no flight should conflict with others when entering the airspace, if all flights follow their initial route. When a D&A algorithm is used, some trajectories may be deviated to avoid conflicts. These modified trajectories may conflict with entering flights, with not enough anticipation for the D&A algorithm to find solutions to these conflicts. Such conflicts on entry are solved by the simulator by delaying the entering flight until it is separated from the other traffic by a distance greater than Δ_{entry}. These delays are measured during the simulations. A large amount of delay, as well as large trajectory deviations, will indirectly result in less flights reaching their destination before the end of the simulation. This will in turn result in a lower traffic throughput than with no D&A. The traffic throughput is measured by the `throughput` metric presented in Section 7.4.

7.3 Experiment Setup and Scenario Parameters

In this paper, some parameters will be the same for all scenarios. The values of these common parameters can be found in Table 1. The aircraft speed is constant during the simulation; only course changes are possible. However, each aircraft might have a different speed. Each aircraft speed is drawn in a uniform distribution $\mathcal{U}([V_{\min}; V_{\max}])$. Different speed ranges will be tested as the speed diversity has a great impact on the observed performance metrics. However, the speed ranges $[V_{\min}; V_{\max}]$ will be chosen so that $(V_{\min} + V_{\max})/2$ is set to a constant value for all scenarios, so that the traffic density is not impacted by the choice of the speed range. This way, the traffic density is only controlled through the average number of flights entering the circular area per unit of time λ. We will use random scenarios with different values of this incoming flow λ in our experiments, as we want to test the algorithms on various traffic densities.

Table 2 gives the values of λ tested in this paper as well as the measures of the traffic density and overall duration of separation loss for each of the three scenarios (low, medium, and high density). These metrics are measured without applying any D&A algorithm in order to characterize the difficulty of each scenario. The cumulated duration of separation loss is simply the sum over all flights of the time during which an aircraft was separated by less than the required minimum separation d from another aircraft.

Parameter	Description	Value
d	Separation standard	5 NM
Δ_{entry}	Radius of exclusion zones preventing conflicts on entry	$4d$
R	Radius of the domain entry	150 NM
α	Random angle range	$[-70°;70°]$
δt	Simulation timestep	5 s
$\frac{V_{\min}+V_{\max}}{2}$	Average speed	230 kts
	Simulation duration inside the stationary regime	6 h

Table 1: Common parameters for all scenarios

Parameter	Description	Low	Medium	High
λ	Incoming flow [flight/h]	60	120	160
	Aircraft density [flight/10 000 NM^2]	8.5	17	22.7
	LoS duration without D&A [s]	60 783.1	244 710.1	432 360.4

Table 2: Low, medium, and high density scenarios. The traffic density is controlled by the value of the incoming flow parameter λ. LoS here means Loss of Separation.

7.4 Performance Metrics

Executing avoidance maneuvers during the simulation of a scenario has an impact on the traffic that can be measured through a number of metrics: number and/or duration of remaining separation losses, delays at entry, trajectory lengthening, number of flights not terminated—i.e. that could not reach their exit point before the end of the simulation because they were maneuvered—and distance remaining to be flown to destination for these flights[4].

In this paper, we propose to assess the performance of D&A algorithms through five metrics only: Two safety metrics measuring the duration and severity of separation losses respectively; Three efficiency metrics, the first measuring the total duration of the degenerate situations with parallel tracks not reaching their destinations, the second measuring the trajectory lengthening due to the D&A maneuvers, and the third the traffic throughput, which is directly related to the amount of delay assigned to entering flights.

Because we want to assess the D&A algorithms performances at a given average traffic density, these safety and efficiency metrics are measured only for the flights that are scheduled to enter the simulation in the time interval corresponding to the stationary regime of the scenario (see Figure 12). So we consider only the flights departing between $t_i^{\text{statio}} = t_i + T_{transition}$ and $t_f^{\text{statio}} = t_f - T_{transition}$ where t_i and t_f are the beginning and end times of the scenario with no detect and avoid, and $T_{transition}$ is the transition time at the beginning or at the end of the scenario. The other flights can be of use when computing some of the metrics, however. For example, a separation loss involving a flight in the selected list and another one that is not in the list will still be counted.

In our experiments, we have taken a transition time $T_{transition} = T_{max} + 2\sigma$ where $T_{max} = 2R\,V_{min}$ and $\sigma = \sqrt{\lambda\,T_{max}}$, in order to be safely within the stationary regime interval.

The performance metrics used to assess the three methods (ORCA, CS-ORCA and DH-ORCA) on scenarios of low, medium and high traffic densities are listed below. The separation loss metrics and the one measuring the parallel tracks duration consider only the trajectory points in the stationary regime of each scenario. The trajectory lengthening metric considers only the trajectory parts starting after the beginning of the stationary regime. Most metrics are normalized so that we can compare their values on scenarios of various traffic densities.

- `normalized_LoS`: This metric quantifies the duration of the separation losses. In order to

[4]Remember that a full simulation with the D&A off will show no delay and have all flights reaching their destination, although it will certainly have many separation losses, whereas with the D&A on we may expect much less separation losses, but some flights might be delayed at entry because of maneuvering aircraft, and some other flights might not reach their destination before the end of the simulation.

avoid scenario-related variations, we normalize this total duration. The `normalized_LoS` metric is the total duration of the separation losses observed in the simulation when using the evaluated D&A algorithm divided by the total duration of separation losses when no D&A algorithm is used. This can be summarized by the following formula, where LoS is the total duration of separation losses:

$$\texttt{normalized\_LoS}(\text{scenario}, \text{algo}) = \frac{\text{LoS}(\text{simu}(\text{scenario}, \text{algo}))}{\text{LoS}(\text{simu}(\text{scenario}, \text{without D\&A}))}$$

- `normalized_LoS_Severity`: This metric quantifies the severity of the separation losses by considering the minimum distance min_LoS_dist between aircraft over all separation losses, and normalizing using the separation standard d (*i.e.* the minimum required separation). This metric is 0 when no separation loss occurs, and is otherwise computed using the following formula:

$$\texttt{normalized\_LoS\_Severity}(\text{scenario}, \text{algo}) = 1 - \frac{\text{min\_LoS\_dist}(\text{simu}(\text{scenario}, \text{algo}))}{d}$$

The closer to 1 this metric is, the worst the closest separation loss is.

- `normalized_length`: With this metric we quantify the trajectory deviations due to the D&A maneuvers. A simple way to do this could be to compute the ratio of the length of the actual trajectories over the length of the initial planned routes. However, for the flights that do not reach their destination before the end of the simulation, it makes no sense to measure the ratio between the length of the incomplete trajectory and the length of the initial planned route. Thus, we consider the distance between the last point of the flown trajectory and the destination, and add it to the length of the flown trajectory, before computing the ratio between this total length and the length of the planned route. This can be summarized by these formulas:

$$\text{length}(\text{scenario}, \text{algo}) = \sum_{\text{traj} \in \text{simu}(\text{scenario}, \text{algo})} \text{length}(\text{traj}) + \|\text{last}(\text{traj}) - \text{dest}(\text{traj})\|$$

$$\texttt{normalized\_length}(\text{scenario}, \text{algo}) = \frac{\text{length}(\text{scenario}, \text{algo})}{\sum_{\text{traj} \in \text{simu}(\text{scenario}, \text{algo})} \|\text{first}(\text{traj}) - \text{dest}(\text{traj})\|}$$

With all aircraft flying straight routes to their destination, this metric would be equal to 1. When aircraft deviate from the straight course, this metric takes values greater than 1. It gives the ratio of the overall trajectory lengths with the D&A active over the overall shortest travel distances from entry to exit that would be flown with the D&A off.

- `throughput`: With this metric we quantify the ability of the D&A to accommodate the incoming flow of aircraft. To measure this, inside the stationary regime, the number of aircraft reaching their destination is compared with the expected theoretical number of aircraft reaching their destination. As we consider a stationary regime, this expected number of aircraft reaching their destination is equal to the expected number of incoming aircraft. This can be summarized by this formula:

$$\texttt{throughput}(\text{scenario}, \text{algo}) = \frac{\text{count\_arrived}(\text{simu}(\text{scenario}, \text{algo}))}{\text{incoming\_flow} \times \text{simulation duration}}$$

If the D&A is not able to accommodate the demand, the amount of delay and the total trajectory lengthening will be high, resulting in a low value of `throughput` (greatly inferior to 1). Please note that this metric can be superior to 1 because the actual number of aircraft reaching their destination can be superior to the expected one. This can be due to the randomness of the Poisson process determining the desired entry times, or it can be due to a large number of aircraft already in the simulation at the beginning of the considered time window in the stationary regime over which the metric is measured.

- `parallel_total_duration`: With this metric we quantify the parallel behavior discussed in Sub-Section 5.3. Two flights at respective positions A and B with velocities $\overrightarrow{v_A}$ and $\overrightarrow{v_B}$ are

considered as locked on parallel tracks when the following conditions are met: $\overrightarrow{v_A}.\overrightarrow{v_B} > 0$ (same direction); the angle between $\overrightarrow{v_A}$ and $\overrightarrow{v_B}$ is less than 5 degrees (parallel tracks); the distance between A and B is less than $1.5\,d$ (proximity) where d is the separation standard; and the intent of each aircraft is to cross path with the other. Denoting $\overrightarrow{v_A}^{\text{pref}}$ and $\overrightarrow{v_B}^{\text{pref}}$ the preferred velocities of the two flights toward their respective destinations, the last criterion (path-crossing) can be expressed as the fact that $\overrightarrow{v_A}^{\text{pref}}$ and B are on the same side of $\overrightarrow{v_A}$, and $\overrightarrow{v_B}^{\text{pref}}$ and A are on the same side of $\overrightarrow{v_B}$.
The `parallel_total_duration` metric is the total duration of the parallel behavior observed in the simulation when using the evaluated D&A algorithm.

7.5 D&A Parameter Selection

Each D&A algorithm has parameters that we must calibrate before comparing the methods: the time horizon τ for ORCA and CS-ORCA and $(\tau_{\text{conflict}}, \tau_{\text{cross}})$ for DH-ORCA. To select the best parameter values for each method, we perform a grid search: The best parameter value (or combination of values) is selected from a grid of values evaluated on a set of scenarios. For each D&A algorithm, each parameter is tested on 3 traffic densities with 20 random scenarios for each traffic density.

Table 3 gives the grid of values tested for each method. For DH-ORCA, note that we try all couples of parameter values such that $\tau_{\text{conflict}} < \tau_{\text{cross}}$.

D&A algorithm	Grid of parameters to be tested
ORCA	$\tau \in \{212\,\text{s}, 236\,\text{s}, \ldots, 740\,\text{s}\}$
CS-ORCA	$\tau \in \{212\,\text{s}, 236\,\text{s}, \ldots, 740\,\text{s}\}$
DH-ORCA	$\tau_{\text{conflict}} \in \{20\,\text{s}, 52\,\text{s}, \ldots, 212\,\text{s}\}$ $\tau_{\text{cross}} \in \{212\,\text{s}, 244\,\text{s}, \ldots, 532\,\text{s}\}$ such that: $\tau_{\text{conflict}} < \tau_{\text{cross}}$

Table 3: Time horizon (τ) values tested for each D&A algorithm.

When choosing the best parameter value(s), one wants to minimize both the separation losses `normalized_LoS` and the trajectory length `normalized_length`, with a higher priority on the safety metric. Considering Figure 13, we see that

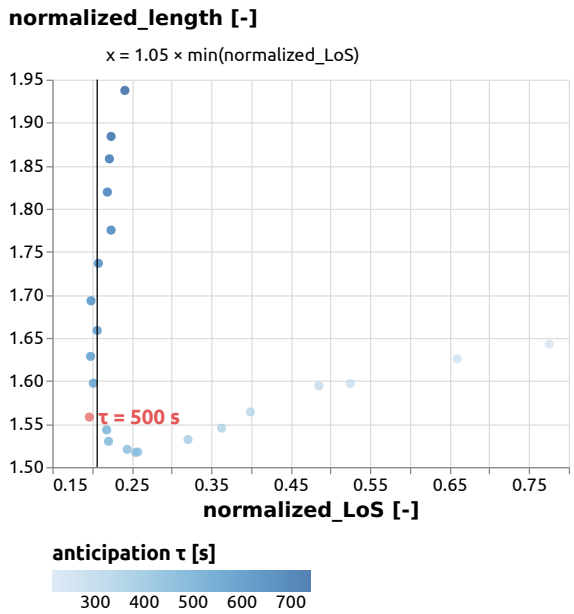


Figure 13: For the ORCA method, `normalized_length` is plotted as a function of the `normalized_LoS`. Each point is the result obtained by using ORCA with a specific τ value. Each point is obtained by computing the metrics average over the 20×3 calibration scenarios. The red point is the selected parameter value $\tau = 500\,\text{s}$ for the ORCA method.

there is a trade-off to be found between the two conflicting objectives (safety and efficiency), when selecting the value of the time horizon parameter τ. This figure plots the normalized trajectory length as a function of the normalized separation loss duration, averaged over the 3×20 calibration scenarios, here for the ORCA method. Each point on this figure represents a parameter value from the selection grid.

As the primary objective is to ensure safety, we could simply select the value of τ minimizing the `normalized_LoS` metric. However, we can see on Figure 13 that there is a range of τ values – near the y-axis – that perform very well in terms of safety, but show different performances in terms of efficiency.

In order to take also the efficiency criteria into account, we will select the τ parameter value having the minimum value of `normalized_length` among all the τ-values for which the `normalized_LoS` measure is less

than 5% above the minimum `normalized_LoS`. This 5% safety threshold is materialized, for ORCA, on Figure 13 by the vertical line close to the y-axis. This selection process is applied for all the time horizon parameters of the three D&A methods tested in this paper (ORCA, CS-ORCA and DH-ORCA).

As stated in section 5.3, when aircraft have to cross one another, a large τ is required to avoid the parallel behavior. In DH-ORCA, we have two horizons, one to avoid the parallel behavior (τ_{cross}) and one to avoid conflict (τ_{conflict}). As a consequence this allows τ_{conflict} to be much smaller than τ_{cross} and the τ values CS-ORCA. This interpretation seems to be confirmed by the best parameter values obtained with the grid search, shown in Table 4.

D&A algorithm	Best performing parameters
ORCA	$\tau = 500\,\text{s}$
CS-ORCA	$\tau = 308\,\text{s}$
DH-ORCA	$\tau_{\text{conflict}} = 52\,\text{s}$ $\tau_{\text{cross}} = 372\,\text{s}$

Table 4: Best values of the time horizon parameter(s) for each D&A algorithm. The τ-values are selected from the parameter grid shown in Table 3, considering the average performance over 3×20 calibration scenarios.

The time horizon parameter values of Table 4 are the ones used in section 8 when comparing the three D&A methods (ORCA, CS-ORCA, DH-ORCA). Please note that the 3×20 scenarios used to select the time horizon parameter values are not used in section 8, where the performances of each method are assessed on new, unseen scenarios, using the metrics presented in section 7.4. This avoids any overly optimistic bias. To draw a parallel with Machine Learning, one can see these 3×20 scenarios as a training set to learn the time horizon parameters. Likewise, the scenarios used in Section 8 can be seen as the test set used to evaluate the performance with new traffic instances.

Note also that the time horizon parameter values found in this section are specific to the context of the current study, and that more realistic traffic conditions – with a different mix of aircraft speeds – might require another selection of these parameters before deploying the chosen D&A algorithm.

8 Results

Let us now compare ORCA, CS-ORCA and DH-ORCA on traffic scenarios of low, medium and high densities (see section 7.3 for a description of the scenarios). In subsection 8.1, we compare the three methods using a same aircraft speed distribution for all the scenarios. Then, in subsection 8.2, we examine the influence of the aircraft speed distribution on the results, considering several speed distributions. As we will see, these two scenario parameters (traffic density, speed distribution) have an impact on the results, although the performance ranking of the three methods remains the same.

For each tested traffic density and speed distribution, 100 random scenarios are simulated. For each simulation, we measure the performance metrics described in section 7.4. The 100 values obtained for each metric are summarized in a boxplot or by their mean value, depending on the considered figure.

The results were computed using only one core of an AMD Ryzen Threadripper 1920X. Computation time is not an issue for geometric D&A algorithms: For instance, it takes 355 s of CPU user time, on average, to simulate 6 hours of traffic using DH-ORCA with the largest incoming flow of 160 flights/hour. Out of these 355 s, the time spent computing the new velocities (*i.e.* the execution time of DH-ORCA) is 262 s.

8.1 Comparison on low, medium, and high traffic density scenarios

In this section, the aircraft speeds in all scenarios are drawn from a same uniform distribution $\mathcal{U}([200\,\text{kts}; 260\,\text{kts}])$.

Let us first see how the ORCA, CS-ORCA and DH-ORCA methods fare in terms of safety, considering the normalized duration of separation losses on Figure 14, and the severity of separation losses on Figure 15. We see that ORCA is the less safe among the three methods, showing the longest and most severe separation losses in all types of scenarios, whereas DH-ORCA outperforms both ORCA and CS-ORCA on these two safety metrics.

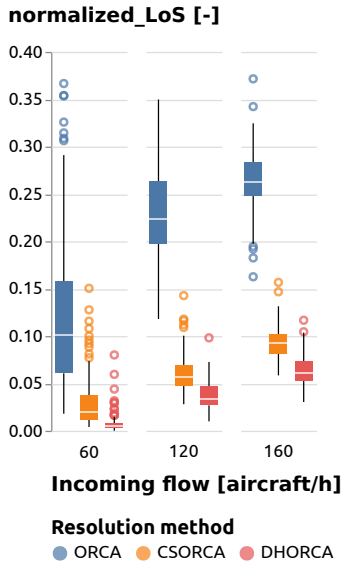


Figure 14: Normalized duration of the separation losses with aircraft speed randomly drawn in $\mathcal{U}$ ([200 kts; 260 kts]) for different incoming flows.

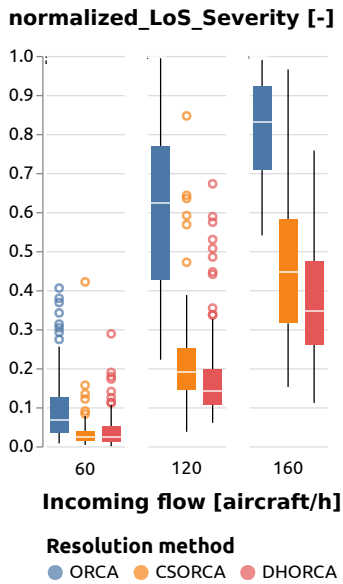


Figure 15: Normalized severity of the separation losses with aircraft speed randomly drawn in $\mathcal{U}$ ([200 kts; 260 kts]) for different incoming flows.

Now considering the efficiency metrics – parallel total duration on Figure 16, normalized trajectory lengthening on Figure 17, and normalized throughput on Figure 18 – we observe similar

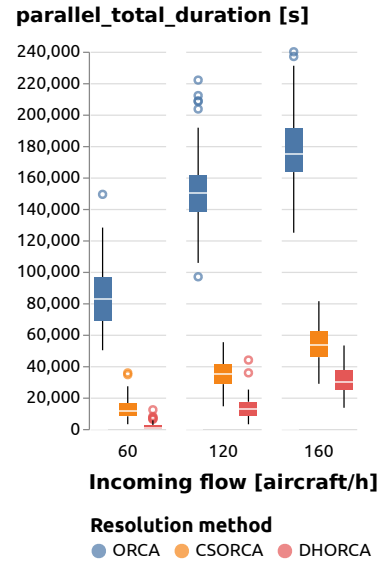


Figure 16: Parallel behaviour total duration with aircraft speed randomly drawn in $\mathcal{U}$ ([200 kts; 260 kts]) for different incoming flows.

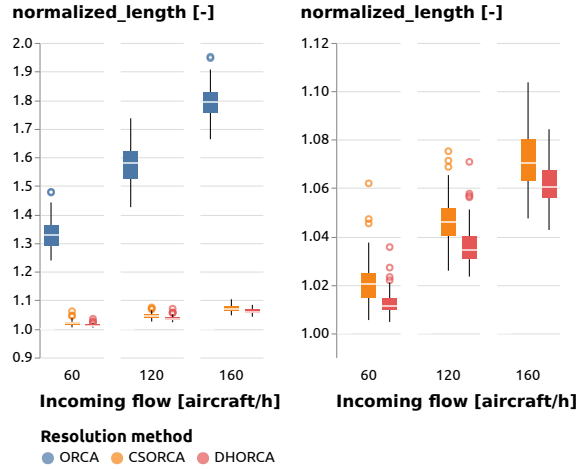


Figure 17: Normalized length with aircraft speed randomly drawn in $\mathcal{U}$ ([200 kts; 260 kts]) for different incoming flows.

results, with ORCA performing worst and DH-ORCA performing best among the three methods.

Figure 16 shows the total duration of the parallel tracks occurrences for the three methods. ORCA performs worst, by far. We also see that CS-ORCA does actually improve very significantly over ORCA on the parallel-tracks issue. DH-ORCA performs best, significantly improving

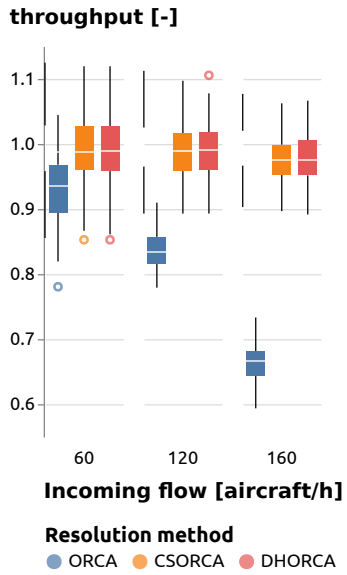


Figure 18: `throughput` with aircraft speed randomly drawn in $\mathcal{U}([200\,\text{kts}; 260\,\text{kts}])$ for different incoming flows.

over both ORCA and DH-ORCA. We will see in the next sub-section that the improvement is even more significant when the speed range is tighter.

The left part of Figure 17 shows that the trajectory lengthening due to the D&A algorithm is really large for ORCA, and much smaller for CS-ORCA and DH-ORCA. This was expected, considering the issues that were raised in section 5.1 concerning ORCA when applied to aircraft flying at constant or near-constant speeds. The right part of Figure 17 zooms on the results of the two best methods, and shows a clear advantage for DH-ORCA.

The better performance of DH-ORCA on both the parallel tracks duration (Fig. 16) and the trajectory lengthening metric (Fig. 17) confirm that the dual horizon actually performs as expected, helping aircraft to cross paths more efficiently than with the two single-horizon methods.

The normalized throughput shown on Figure 18 is closely related both to the additional time spent by aircraft in the area due to maneuvering, and to the delays assigned to entering aircraft, due to other traffic maneuvering close to their entry point. Remind that the scenarios are designed such that no conflict occurs at entry when no D&A algorithm is used, but that delays may be assigned when a D&A algorithm modifies the trajectories. When the cumulated delays are important, this reduces the throughput of the system. We see on Figure 18 that both CS-ORCA and DH-ORCA are able to maintain a nearly constant throughput ratio, whatever the traffic density, whereas the throughput drastically decreases with the traffic density (which is controlled through the incoming flow) when ORCA is used.

8.2 Influence of the aircraft speed distribution

In all the scenarios in this paper, every aircraft flies at a constant speed: only course changes are possible. The individual aircraft speeds are drawn in a uniform distribution centered on a same average speed (see Table 1 in section 7.3). In the current section, we investigate the influence of the speed range of the uniform distribution on the results.

As seen in Section 5.3, it may happen that the D&A algorithm solves a conflict by postponing it, setting the two involved aircraft on parallel courses. If the aircraft fly at different speeds, one will eventually overtake the other, and both aircraft will resume their navigation towards their respective destination. However, if they fly at the same speed, the relative position of the aircraft will stay the same: they are interlocked. This situation is more likely to occur when the speed diversity is low.

Let us now confirm that DH-ORCA actually mitigates this issue. When comparing DH-ORCA with CS-ORCA on a variety of speed ranges, DH-ORCA should perform better than CS-ORCA when the speed diversity is low. This is actually the case, as can be seen on Figures 19, 20, 21, 22, and 23 showing the evolution of the metrics with the speed range. In these figures, the mean values of the metrics are plotted along with 95% confidence intervals for the estimated means.

Figures 19 and 20 show that the trajectory lengthening and the parallel behavior total duration remain approximately constant with DH-ORCA, whatever the speed range, whereas with CS-ORCA they both dramatically increases when the speed diversity is low. Likewise, the throughput in Figure 21 is stable for DH-ORCA whereas it slightly drops for CS-ORCA as speed diversity decreases.

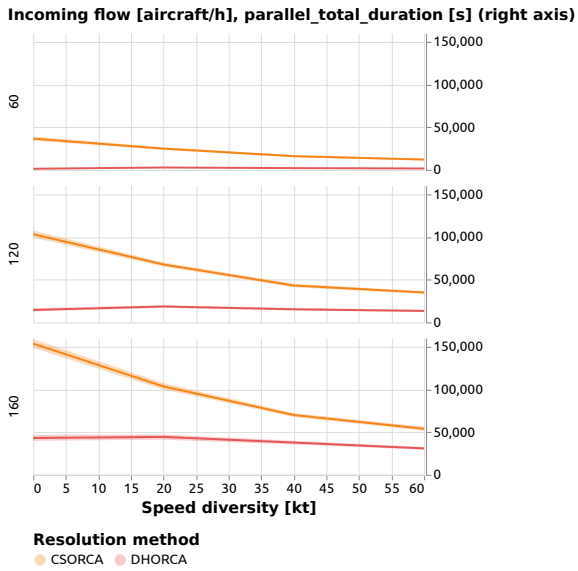


Figure 19: Evolution of the parallel behaviour total duration with the speed range.

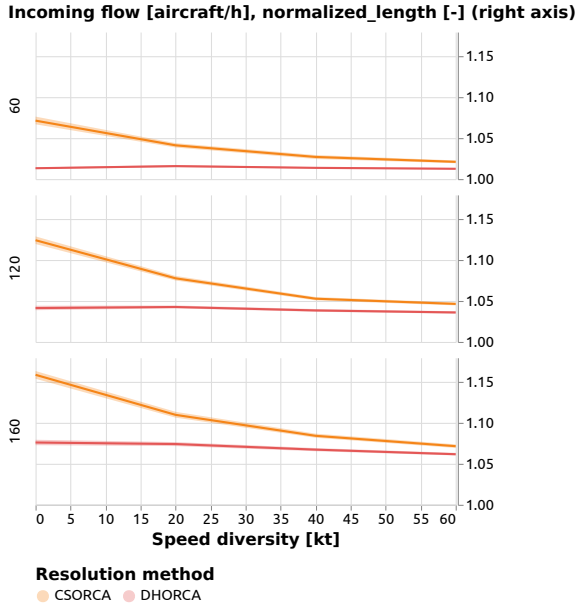


Figure 20: Evolution of the normalized trajectory lengthening with the speed range.

We can conclude that he pathological behavior for aircraft flying at similar constant speeds—*i.e.* aircraft being set on parallel tracks—that occurred when using ORCA is still observed with CS-ORCA and the more so with low speed diversity. Although it still exists with DH-ORCA, this

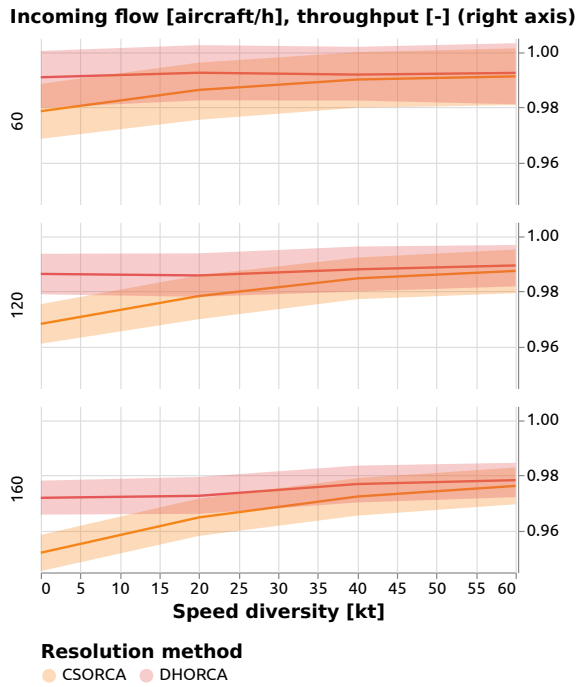


Figure 21: Evolution of the normalized throughput with the speed range.

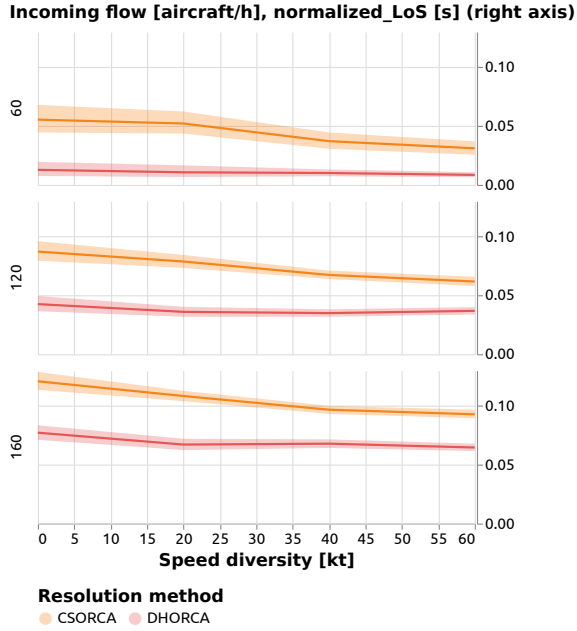


Figure 22: Evolution of the normalized duration of separation losses with the speed range.

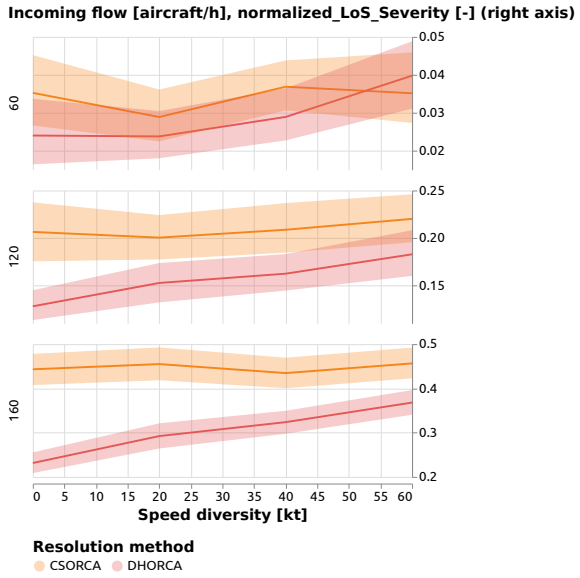


Figure 23: Evolution of the normalized severity of the separation losses with the speed range.

pathological behavior is significantly mitigated by the use of a dual horizon.

Concerning the safety metrics (duration of separation losses and separation loss severity), we see on Figures 22 and 23 that DH-ORCA consistently fares better than CS-ORCA on all traffic densities and speed ranges. On Figure 22, we can also observe that DH-ORCA is somewhat less sensitive to the speed diversity than CS-ORCA, in terms of conflict duration.

Looking at Figure 23, we can see that the remaining separation losses are less severe with DH-ORCA than with CS-ORCA for the `normalized_LoS_Severity` metric. Considering the evolution of the severity with the speed range, we see that the advantage of using DH-ORCA is even more apparent when aircraft fly at very similar speeds.

9 Conclusion

To conclude, we have seen that the original Optimal Reciprocal Collision Avoidance (ORCA) algorithm, initially proposed in van den Berg et al (2011) for robot collision avoidance, exhibits a pathological behavior when applied for Detect & Avoid purposes to aircraft or UAS flying at similar constant speeds. The Constant-Speed ORCA (CS-ORCA) variant of the algorithm introduced in Durand (2018) was designed as a way to solve this issue.

In the current paper, we have exposed why CS-ORCA might still exhibit the undesired behavior where two flights converging at a small angle might be set on parallel tracks by the algorithm, preventing them to cross paths to reach their respective destinations. We have shown that CS-ORCA is less subject to this issue when using a large value of the time horizon parameter τ. However, our results in section 8.2 show that this issue still remains when using CS-ORCA, even though we selected the τ value in section 7.5 to get the best possible results on the kind of scenarios considered in this study.

We have proposed a new algorithm Dual-Horizon ORCA (DH-ORCA) combining a short-term ORCA-like logic with a longer-term CS-ORCA-like logic with optional constraints. In DH-ORCA, a relatively small time horizon τ_{conflict} is used to enforce separation, and a larger time horizon τ_{cross} is used to help aircraft (and/or UAVs) to cross paths. The optional constraints computed with the larger time horizon τ_{cross} are designed so that each aircraft only requires its own intent to decide to enforce the constraint or not: There is no need for any aircraft to broadcast its destination to the others.

Our results show that DH-ORCA outperforms both ORCA and CS-ORCA on scenarios of various traffic densities. They also show that the gain in safety and efficiency of DH-ORCA is most important when the speed diversity is low, as expected.

The results of our study were obtained on traffic scenarios where all aircraft fly at similar, constant speeds, in order to make the undesired behavior described above more likely to occur. Aircraft flying at similar speeds within a very limited speed range is actually a common feature of the commercial traffic in the upper airspace, although these aircraft usually fly at greater speeds than in our study. In the lower airspace, the air traffic that we may envision in the future will probably also include a number of UAVs and/or aircraft operating at similar speeds—closer to the speed ranges of our scenarios—and for which a speed modification is either of very limited range (e.g., large fixed-wing UAVs) or costly in terms of energy budget (e.g., multi-rotor or hybrid UAVs).

So the Detect & Avoid logic that could be deployed in the future on these aircraft or UAVs should be safe and efficient in all situations, including when aircraft and/or UAVs flying at similar speeds converge at a small angle. We have shown that DH-ORCA can fulfill this objective with greater safety and efficiency than ORCA or CS-ORCA.

However, none of the geometric algorithms studied in this paper provides a formal guarantee to avoid the parallel tracks degenerate situations. Although they are much less likely to occur with DH-ORCA than with ORCA or CS-ORCA, these situations still happen in dense traffic and we cannot be sure they will never occur even with low traffic.

Further research could focus on finding an algorithm that would provide such formal guarantees. One could also examine the robustness of geometric algorithms such as ORCA or its variants CS-ORCA and DH-ORCA, and possibly other D&A logics, in more realistic operational conditions where all aircraft and/or UAVs might not share the exact same representation of the traffic, and might have slightly different implementations of the Detect & Avoid logic.

Declarations

- **Funding** The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.
- **Conflict of Interest/Competing Interests** The authors have no financial or proprietary interests in any material discussed in this article.
- **Data Availability** Not applicable, as no real data have been used to realize this work.
- **Code availability** The custom code used during the current study is available from the corresponding author on reasonable request.
- **Authors' Contributions** Richard Alligier: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. David Gianazza: Conceptualization, Methodology, Software, Writing – Original Draft, Writing – Review & Editing, Supervision. Nicolas Durand: Conceptualization, Methodology, Supervision, Writing – original draft, Writing – review & editing. Xavier Olive: Conceptualization, Visualization, Writing – original draft, Writing – review & editing.
- **Ethics Approval** Not applicable.
- **Consent to Participate** Not applicable.
- **Consent to Publish** Not applicable.

References

Alligier R, Allignol C, Barnier N, et al (2018) Detect and avoid algorithm for uas with 3d-maneuvers. In: ICRAT 2018, 8th International Conference on Research in Air Transportation

Allignol C, Barnier N, Flener P, et al (2012) Constraint programming for air traffic management: a survey. The Knowledge Engineering Review 27(03):361–392. https://doi.org/10.1017/S0269888912000215, URL http://www.journals.cambridge.org/abstract_S0269888912000215$

Allignol C, Barnier N, Durand N, et al (2013) A new framework for solving en-route conflicts. In: Proceedings of the 10th ATM R&D Seminar, Chicago, IL

Alliot JM, Durand N, Granger G (2001) A statistical analysis of the influence of vertical and ground speed errors on conflict probe. In: ATM 2001, 4th USA/Europe Air Traffic Management Research and Development Seminar, Santa Fe, United States, URL https://hal-enac.archives-ouvertes.fr/hal-00938005

Alonso-Ayuso A, Escudero LF, Martin-Campo FJ (2016) Exact and approximate solving of the aircraft collision resolution problem via turn changes. Transportation Science 50(1):263–274. https://doi.org/10.1287/trsc.2014.0557

Archambault N, Durand N (2004) Scheduling heuristics for on-board sequential air conflict solving. In: 23rd Digital Avionics Systems Conference, AIAA, DASC 2004 Proceedings, vol 1. IEEE, Salt Lake City, UT, pp 3.1–9

Balasooriyan S (2017) Multi-aircraft conflict resolution using velocity obstacles. Master's thesis, TUDelft

van den Berg J, Lin M, Manocha D (2008) Reciprocal velocity obstacles for real-time multi-agent navigation. In: 2008 IEEE International Conference on Robotics and Automation, pp 1928–1935, https://doi.org/10.1109/ROBOT.2008.4543489

van den Berg J, Guy SJ, Lin M, et al (2011) Reciprocal n-body collision avoidance. In: Pradalier C, Siegwart R, Hirzinger G (eds) Robotics Research: The 14th International Symposium ISRR, Springer Tracts in Advanced Robotics (STAR), vol 70. Springer, Berlin, Heidelberg, pp 3–19

Bilimoria K (2000) A geometric optimization approach to aircraft conflict resolution. In: Proceedings of the 18th Applied Aerodynamics Conference

Bonini D, Dupré C, Granger G (2009) How erasmus can support an increase in capacity in 2020. In: Proceedings of the 7th International Conference on Computing, Communications and Control Technologies: CCCT 2009, Orlando, Florida

Bulusu V, Sengupta R, Liu Z (2016) Unmanned aviation: To be free or not to be free? a complexity based approach. In: 7th International Conference on Research in Air Transportation, Drexel University, Philadelphia, PA, ICRAT 2016 Proceedings

Chiang YJ, Klosowski JT, Lee C, et al (1997) Geometric algorithms for conflict detection/resolution in air traffic management. In: Proceedings of the 36th Conference on Decision and Control, San Diego, CA, pp 1835–1840

Christodoulou MA, Kontogeorgou C (2008) Collision avoidance in commercial aircraft free flight via neural networks and non-linear programming. International Journal of Neural Systems 18(5):371–387

D. Bilimoria K, Sridhar B, B. Chatterji G, et al (2001) Facet: Future atm concepts evaluation tool. Air Traffic Control Quarterly 9:1–20

D'Amato E, Mattei M, Notaro I (2020) Distributed reactive model predictive control for collision avoidance of unmanned aerial vehicles in civil airspace. Journal of Intelligent & Robotic Systems 97. https://doi.org/10.1007/s10846-019-01047-5

d'Engelbronner J, Borst C, Ellerbroek J, et al (2015) Solution-space-based analysis of dynamic air traffic controller workload. Journal of Aircraft pp 1146–1161

Durand N (2018) Constant speed optimal reciprocal collision avoidance. Transportation Research Part C: Emerging Technologies 96:366–379. https://doi.org/https://doi.org/10.1016/j.trc.2018.10.004, URL https://www.sciencedirect.com/science/article/pii/S0968090X18314232

Durand N, Barnier N (2015) Does atm need centralized coordination? autonomous conflict resolution analysis in a constrained speed environment. In: 11th USA/Europe Air Traffic Management Research and Developpment Seminar

Durand N, Alliot JM, Noailles J (1996) Automatic aircraft conflict resolution using genetic algorithms. In: 11th Annual Symposium on Applied Computing, ACM, Philadelphia, PA, pp 289–298

Eby MS, Kelly WEIII (1999) Free flight separation assurance using distributed algorithms. In: IEEE Aerospace Conference. Proceedings, pp 429–441

Emmanuel Sunil JE, Hoekstra JM (2018) Camda: Capacity assessment method for decentralized air traffic control. In: 8th International Conference on Research in Air Transportation, UPC, Castelldelfels, Spain, ICRAT 2018 Proceedings

Frazzoli E, Mao ZH, Oh JH, et al (2001) Resolution of conflicts involving many aircraft via semidefinite programming. Journal of Guidance, Control and Dynamics 24(1):79–86

Gariel M, Feron E (2009) 3d conflict avoidance under uncertainties. In: 28th Digital Avionics Systems Conference, AIAA. IEEE, Orlando, FL, DASC 2009 Proceedings, pp 4.E.3–1–4.E.3–8

Granger G, Durand N (2003) A traffic complexity approach through cluster analysis. In: ATM 2003, 5th USA/Europe Air Traffic Management Research and Development Seminar, Budapest, Hungary, p pp xxxx, URL https://hal-enac.archives-ouvertes.fr/hal-00938044

Granger G, Durand N, Alliot JM (2001a) Optimal resolution of en-route conflicts. In: Proceedings of the 4th ATM R&D Seminar, Santa Fe, NM

Granger G, Durand N, Alliot JM (2001b) Token allocation strategy for free-flight conflict solving. In: Hirsh H, Chien S (eds) IAAI 2001, 13th Conference on Innovative Applications of Artificial Intelligence. AAAI Press, Seattle, WA, pp 59–64

Guo K, Wang D, Fan T, et al (2021) Vr-orca: Variable responsibility optimal reciprocal collision avoidance. IEEE Robotics and Automation Letters 6(3):4520–4527

Hoekstra J, van Gent R, Ruigrok R (2002) Designing for safety: the 'free flight' air traffic management concept. Reliability Engineering & System Safety 75(2):215 – 232. https://doi.org/https://doi.org/10.1016/S0951-8320(01)00096-5, URL http://www.sciencedirect.com/science/article/pii/S0951832001000965

Hu J, Prandini M, Nilim A, et al (2002) Optimal coordinated maneuvers for three dimensional aircraft conflict resolution. Journal of Guidance, Control and Dynamics 25(5):888–900

Košecká J, Tomlin C, Pappas GJ, et al (1998) 2 1/2 d conflict resolution maneuvers for atms. In: Proceedings of the 37th IEEE Conference on Decision and Control, pp 2650–2655

Krella F, et al (1989) Arc 2000 scenario (version 4.3). Tech. rep., Eurocontrol

Le Ny J, Pappas GJ (2010) Geometric programming and mechanism design for air traffic conflict resolution. In: Proceedings of the 2010 American Control Conference (ACC), pp 3069–3074

Lehouillier T, Omer J, Soumis F, et al (2017) 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. https://doi.org/https://doi.org/10.1016/j.ejor.2016.07.008, URL http://www.sciencedirect.com/science/article/pii/S0377221716305458

Oh JH, Shewchun JM, Feron E (1997) Design and analysis of conflict resolution algorithms via positive semidefinite programming [aircraft conflict resolution]. In: Proceedings of the 36th Conference on Decision and Control, San Diego, CA, pp 4179–4185

Omer J, Farges JL (2013) Hybridization of nonlinear and mixed-integer linear programming for aircraft separation with trajectory recovery. IEEE Transactions on Intelligent Transportation Systems 14(3):1218–1230. https://doi.org/10.1109/TITS.2013.2257758

Pallottino L, Bicchi A, Feron E (2001) Mixed integer programming for aircraft conflict resolution. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, Montréal, Canada

Pallottino L, Feron E, Bicchi A (2002) Conflict resolution problems for air traffic management systems solved with mixed integer programming. IEEE Transactions on Intelligent Transportation Systems 3(1):3–11

Pallottino L, Scordio VG, Bicchi A, et al (2007) Decentralized cooperative policy for conflict resolution in multivehicle systems. Trans Rob 23(6):1170–1183. https://doi.org/10.1109/TRO.2007.909810, URL http://dx.doi.org/10.1109/TRO.2007.909810

Peyronne C, Conn A, Mongeau M, et al (2015) Solving air-traffic conflict problems via local continuous optimization. European Journal of Operational Research 241(2):502–512. https://doi.org/10.1016/j.ejor.2014.08.045, URL https://hal-enac.archives-ouvertes.fr/hal-00912785

Rey D, Hijazi H (2017) Complex number formulation and convex relaxations for aircraft conflict resolution. In: 2017 IEEE 56th Annual

Conference on Decision and Control (CDC), Melbourne, Australia, pp 88–93

Ruigrok RC, Hoekstra JM (2007) Human factors evaluations of free flight: Issues solved and issues remaining. Applied Ergonomics 38(4):437 – 455. https://doi.org/https://doi.org/10.1016/j.apergo.2007.01.006, URL http://www.sciencedirect.com/science/article/pii/S000368700700018X, flightdeck of the Future

Schouwenaars T, Feron E (2004) Decentralized cooperative trajectory planning of multiple aircraft with hard safety guarantees. In: AIAA Guidance, Navigation, and Control Conference and Exhibit

Snape J, Manocha D (2010) Navigating multiple simple-airplanes in 3D workspace. In: IEEE International Conference on Robotics and Automation (ICRA), Anchorage, AK, pp 3974–3980

Sunil E, Ellerbroek J, Hoekstra J, et al (2017) Modeling airspace stability and capacity for decentralized separation. In: 12th USA/Europe Air Traffic Management Research and Development Seminar, Seattle, United States

Wang R, Alligier R, Allignol C, et al (2020) Cooperation of combinatorial solvers for en-route conflict resolution. Transportation research Part C, Emerging technologies 114:36–58. https://doi.org/10.1016/j.trc.2020.01.004, URL https://hal-enac.archives-ouvertes.fr/hal-02733455

Wang S, Hu X, Xiao J, et al (2021) Repulsion-oriented reciprocal collision avoidance for multiple mobile robots. Journal of Intelligent & Robotic Systems 104(1):12. https://doi.org/10.1007/s10846-021-01528-6, URL https://doi.org/10.1007/s10846-021-01528-6

Zeghal K (1998) 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, pp 273–278

Richard Alligier received his Ph.D. (2014) degree in Computer Science from the "Institut National Polytechnique de Toulouse" (INPT), his engineer's degree (IEEAC, 2010) from the french university of civil aviation (ENAC) and his M.Sc. (2010) in computer science from the University of Toulouse. He is currently assistant professor at the ENAC in Toulouse, France. His research interests include machine learning and optimization methods applied to aircraft trajectory prediction and aircraft conflict resolution.

David Gianazza is currently associate professor at the "École Nationale de l'Aviation Civile" (ENAC), Toulouse. He is also associate researcher in the Parallel Algorithms and Optimization team of the Institut de Recherche en Informatique de Toulouse (IRIT) since 2008. He received his engineer degrees (1986, 1996) from ENAC and his M.Sc. (1996) and Ph.D. (2004) in Computer Science from the "Institut National Polytechnique de Toulouse" (INPT). He obtained his HDR (a french post-doctoral degree similar to tenure) in 2016 from the INPT. He has held various positions in the french civil aviation administration, successively as an engineer in Air Traffic Control operations, technical manager, and researcher. His research focuses mainly on the application of Artificial Intelligence methods to air traffic management problems.

Nicolas Durand is a professor at the École Nationale de l'Aviation Civile (ENAC). He graduated from the École Polytechnique de Paris in 1990 and from ENAC in 1992. He has been a design engineer at the Centre d'Études de la Navigation Aérienne (then DSNA/DTI R&D) from 1992 to 2008, holds a PhD in Computer Science (1996) and got his HDR (French tenure) in 2004.

Xavier Olive is a full researcher with ONERA, the French Aerospace Lab. He graduated from Supaero, Université de Toulouse, France and holds a PhD from Kyoto University, Japan. His research interests include Data Science, Machine Learning and Decision Science applied to aviation, with a particular focus on optimisation, anomaly, and pattern detection applied to air traffic management, operations, predictive maintenance, safety analyses and risk assessment.