Development of a Framework to Compare Low-Altitude Unmanned Air Traffic Management Systems

Coline Ramée* and Dimitri Mavris†
Aerospace Systems Design Laboratory, School of Aerospace Engineering,
Georgia Institute of Technology, Atlanta, GA, 30332, USA

Several reports forecast a very high demand for Urban Air Mobility services such as package delivery and air taxi. This would lead to very dense low-altitude operations which cannot be safely accommodated by the current air traffic management system. Many different architectures for low-altitude air traffic management have been proposed in the literature, however, the lack of a common framework makes it difficult to compare strategies. The work presented here establishes efficiency, safety and capacity metrics, defines the components of an automated traffic management system architecture and introduces a preliminary framework to compare different alternatives. This common framework allows for the evaluation and comparison of different alternatives for unmanned traffic management. The framework is showcased on different strategies with different architectures. The impact of algorithmic choices and airspace architectures is evaluated. A decoupled approach to 4D trajectory planning is shown to scale poorly with agents density. The impact of segregating traffic by heading is shown to be very different depending on the algorithms and airspace access rules chosen.

I. Nomenclature

$4DT$	=	4-Dimensional Trajectory
ATM	=	Air Traffic Management
HIP	=	Horizontal Intrusion Parameter
LoS	=	Loss of Separation
$NMAC$	=	Near Midair Collision
UAM	=	Urban Air Mobility
UAS	=	Unmanned Aircraft System
UTM	=	UAS Traffic Management

II. Introduction

NASA defines Urban Air Mobility (UAM) as a "safe and efficient system for air passenger and cargo transportation within an urban area, inclusive of small package delivery and other urban Unmanned Aerial Systems (UAS) services, which supports a mix of onboard/ground-piloted and increasingly autonomous operations" [1]. Both package delivery drones and larger passenger-carrying eVTOLs have generated a lot of interest from industry [2–4]. The forecast traffic in some reports far exceeds the current capabilities of the US Air Traffic Management System [5, 6]. Moreover, NASA and the FAA have stated that a UAM traffic management system should be "safely integrating operations without burdening current ATM" [7]. Hence, a new separate traffic management system to coordinate high densities of flights at low-altitude must be designed.

To design a new system, a systematic decision-making process should be followed in order to maximize performance. Decision-making process at the conceptual design stage should follow these steps: (1) establish the need, (2) define the problem, (3) establish value objectives, (4) generate feasible alternatives, (5) evaluate alternatives, and (6) make decision [8].

Several papers have been published which propose different alternatives for organizing low-altitude air traffic [9–16]. However, every study defines the problem slightly differently and proposes different value objectives or metrics.

*Graduate Research Assistant, AIAA student member
†S.P. Langley Distinguished Regents Professor, AIAA Fellow.

Moreover, with a few exceptions, these papers each propose one alternative. There is no systematic approach to generating all feasible alternatives as would be required by a rigorous decision-making process. Some alternatives remain at a Concept of Operations (ConOps) level and have not properly been evaluated. Other alternatives have been evaluated using simulation environments, but each concept is evaluated with different simulators that may or may not be publicly available. This makes it difficult to evaluate and compare the alternatives.

To address these gaps and allow a better decision-making process for Unmanned Traffic Management (UTM), we first propose a simple ontology for UTM which allows for alternatives to be generated systematically. We show how alternatives from the literature fit into this ontology. Then we introduce a framework consisting of metrics and a simulation environment that can be used to evaluate UTM alternatives. We show the value of the framework with two studies evaluating different alternatives. The first study compares the performance of different free airspace alternatives. The second study evaluates the impact of different airspace structures on the alternatives.

Some ConOps, such as the one proposed by NASA, use a centralized authority to ensure that vehicles' trajectories do not conflict with each other [17]. However, there is no information as to how to plan a conflict-free trajectory using centralized information about other vehicles' flight plans in an automated manner. In this paper three methods to plan conflict-free 4D trajectories are proposed and evaluated: a method that decouples path planning and timing, an algorithm from the robotics community named Safe Interval Path Planning (SIPP), and an algorithm which uses a local collision avoidance method to plan its path. These algorithms are presented in details in section IV.B.2.

III. System Decomposition and Previous Work

A. System Decomposition

For manned ATM an ontology for the traffic management system was presented in [18]. Keller introduces a breakdown of the system in the following subsystems: (1) airspace components (sectors, routes, etc.), (2) departure and arrival routes, (3) flights (flight plans and flight paths), (4) control facilities (towers, air route traffic control centers, etc.), (5) traffic management initiatives (ground delay programs, miles-in-trail, flow constrained areas, etc.), (6) airlines, (7) aircraft (vehicle type and performance), (8) infrastructure (airports and their subsystems such as terminal, gates, runways, etc.), and (9) weather. This was developed for data modeling of the existing ATM system, but such a decomposition would be useful to generate and classify feasible alternatives for the unmanned traffic management system.

However, this ontology is not directly applicable to generate alternatives for UTM systems. First, it is tailored to the current commercial air traffic management system and some subsystems such as traffic management initiatives are very specific. Second, it has a high level of detail which is not manageable for a conceptual design step of a system-of-system. Third, it mixes subsystems that can be designed by the decision-maker, such as airspace components or traffic management initiatives, with subsystems that are outside of the system designer control, such as weather or airlines.

This is why we propose a simple decomposition of UTM systems into four subsystems, loosely based on the ATM ontology, to generate alternatives:

- Airspace structure,
- Access control,
- Preflight planning, and
- Collision avoidance.

The infrastructure, weather and airlines subsystems are not modeled in the preliminary framework and only a simple model for the aircraft is implemented in order to keep the complexity manageable at the conceptual design stage.

The following subsections explain in more details each of the different subsystems, and provide examples of possible implementation of these subsystems in a UTM context. Table 1 shows how proposed alternatives from the literature fit in this decomposition.

1. Airspace Structure

We define airspace Structure as the rules that regulate airspace navigation. There can be many different types of structures:

- Free-Flight: Aircrafts can fly their preferred path to their destination (also called free routing)
- Routes: Aircrafts can be constrained to fly on a network of airways defined by specific waypoints
- Layers: Aircrafts altitude can be constrained based on the type of operation and/or direction of travel

- Sectors: The airspace is divided into areas that can restrict access or impose additional rules

The current National Airspace is a mix of all these structures. Papers that proposed alternatives for unmanned air traffic management have often focused on a single airspace structure, with the notable exception of the Metropolis project [9]. In [19], it was shown that using layers would reduce the probability of a conflict for manned commercial traffic. Similar results where shown in [9] and [14] for Urban Air Mobility applications. Heavily structured airspace such as routes and sectors were shown on the contrary to increase local density and result in more conflicts.

2. Access Control

Access Control is the system that regulates traffic flow by enforcing who can access sections of the airspace. Current VFR traffic in uncontrolled airspace operates without access control, any aircraft can take-off if there are no immediate conflicts. For commercial traffic ATC can use capacity to regulate traffic in sectors in order to keep workload manageable for air traffic controllers. In the NASA UTM access control is regulated by reserving volumes of airspace, the protected volume is called a geofence. The volume is reserved to a single vehicle from take-off to landing. An alternative is to have the protected volume dynamically follow the vehicle. This is also known as 4-Dimensional Trajectory (4DT). In 4DT, an aircraft flight plan not only states its trajectory in space but also in time, which would allow to coordinate trajectories strategically rather than reactively, while keeping the blocked volume of airspace to a minimum. 4DT is a key element of Trajectory Based Operations (TBO) which is being studied as part of the FAA NextGen initiative [20]. If the 4DT trajectory is perturbed beyond its tolerance due to disturbances or disrupted by higher priority or non-compliant vehicles, it would have to either try to re-plan or fallback to a decentralized approach [13].

3. Preflight Planning

Preflight Planning is the algorithms used to strategically plan a path before take-off. If there is an access control system then there must be rules and algorithms in place for agents to request access.

Planning a conflict-free trajectory in 4D for many agents is computationally expensive. To reduce the cost associated with trajectory planning, some papers have proposed a decoupled strategy in which the spatial path is first determined and then the velocity required to avoid conflicts along that path is determined. In [10], the 3D path is fixed to be the shortest path and a ground delay is applied until the path is clear of conflict. In [11], the vehicle's initial strategy is a straight path at the maximum altitude and the paper compares and combines three heuristics to perform conflict resolution. Two of them act on ground delays while the third considers local avoidance maneuvers. In [13], the authors fix the 3D path and propose to use simulated annealing to find a ground delay that minimizes a weighted sum of conflict duration and delays. In [13] and [11], the analysis is done for scheduled flights, which allow to optimize the overall traffic as opposed to just one flight.

4. Collision Avoidance

Collision Avoidance is the detect and avoid systems and algorithms on-board the vehicle. Collision avoidance methods are reactive methods, the agent's path is updated as it goes along based on the surrounding traffic. Reactive strategies can be :

- Centralized: agents are in communication with a central authority which is in charge of providing deconfliction capabilities. The current ATC system for IFR flight is centralized.
- Decentralized: agents follow sets of rules to avoid each other with only basic information about surrounding traffic (velocity, position). In [15], the authors compare several decentralized collision avoidance methods.
- Collaborative: agents can communicate to decide on avoidance maneuvers. For instance on commercial airplanes the TCAS systems of the two aircraft involved in the conflict communicate to coordinate their resolution advisories [21]

B. UTM Alternatives proposed in the Literature

In this section, we show how alternatives that have been proposed in past papers fit the proposed system decomposition. Table 1 lists some of these alternatives and the simulation framework used for the study if one was used. Note that in this table a preflight planning of none means that traffic is not considered (there is no access control). In that case, the preflight planning is usually limited to finding the shortest path, usually a straight line to the goal. However, in the Zone concept the airspace structure strongly constrains the trajectory and requires a static planning algorithm (the well-known shortest path on a graph algorithm A* is used).

Table 1 Proposed Alternatives for Unmanned Traffic Management System in Literature

Architecture	Airspace Structure	Access Control	Preflight Planning	Collision Avoidance	Simulation Framework
Full Mix [9]	Free	None	None	Decentralized reactive (MVP)	TMX
Layers [9]	Layers	None	None	Decentralized reactive (MVP)	TMX
Zones [9]	Sectors with prescribed heading range	None	None	Decentralized reactive (MVP)	TMX
Tubes [9]	Route Network	4DT contract at the nodes	Centralized A* to find first available route	None	TMX
NASA UTM [17]	Free	Volume	Volume Reservation (manual)	None	N/A
Iowa UTM [10]	Free	Volume	Static A*	None	Custom 2D
Sky Highways [12]	Routes	None	None	1D velocity adjustment	Custom 1D
DLR Delivery Network [11]	Free	4DT contract	Decoupled Scheduled Operations (ground delay heuristics, altitude maneuver)	None	Custom 3D
Onera VLL Operations [13]	Free	4DT contract	Decoupled with ground delays	Swarm Algorithm	N/A
Altiscope [14]	Free	4DT	Local collision avoidance maneuver	Static iterative method	Custom 2D
Ames Reactive [15]	Free	None	None	ICAROUS-based, hover, potential field	Custom and Fe3
Linköping distributed [16]	Layers	None	None	Hovering, Layer Change	Custom 3D
Linköping centralized [16]	Layers	4DT Contract	Ground delay, layer assignment	None	Custom 3D

From this table it can be seen that there have been many different alternatives proposed but that there is a lack of a common framework or simulator to compare them. Many simulators were custom built for the study and were not released. The TMX simulator is not publicly available. Fe3 has not yet been released for public use. The Bluesky simulator is still in active development and appears to be primarily focused on airplanes and free-flight operations and as a result would require significant modifications [22].

There is a need for a framework that would allow the 4 different subsystems to be varied and compared in terms of the same safety, efficiency and capacity metrics under the same assumptions.

IV. Proposed Framework

This section goes into more details into each element of the framework. As illustrated on Figure 1, the different elements are: (1) System decomposition, (2) Generate Alternatives, (3) Establish Value Objectives and (4) Evaluate alternatives. The system decomposition was presented in the previous section. In this section, first the different metrics used as value objectives are presented. Then, the simulation used to evaluate alternatives and the subsystems that were implemented to generate new alternatives are presented.

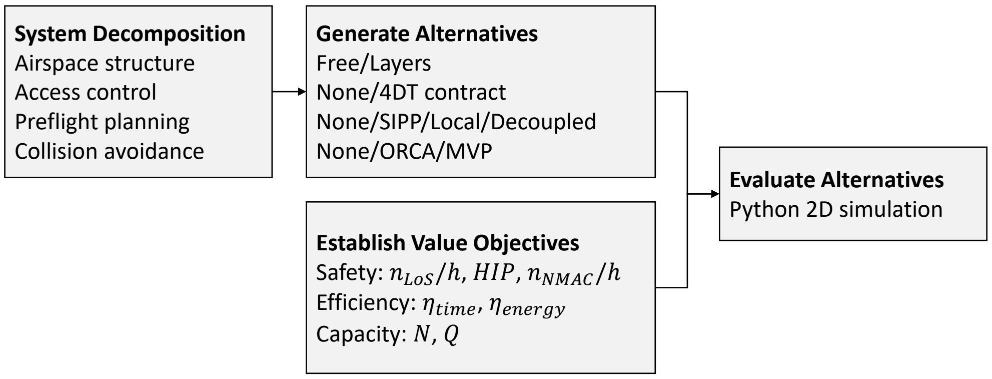

Fig. 1 Framework Graphical Summary

A. Metrics

To compare alternatives a common set of metrics must be defined which will allow alternatives to be ranked in terms of safety, efficiency and capacity.

1. Efficiency

Efficiency can be defined with respect to time or energy, and alternatives leading to good energy efficiency may not lead to good time efficiency. For example, the energy efficiency is maximized in [10] by always flying the shortest path, but this can result in long ground delays for the vehicle. In other instances, neither time nor energy is directly minimized. In the tube topology proposed in [9], the ground delay is minimized by selecting the first available path found by the shortest path algorithm even though it might result in a path that is longer and might even arrive later than a delayed start. In free flight approaches, the initial planned path is the shortest path and local changes are made in case of conflicts, so the paths are not optimized globally.

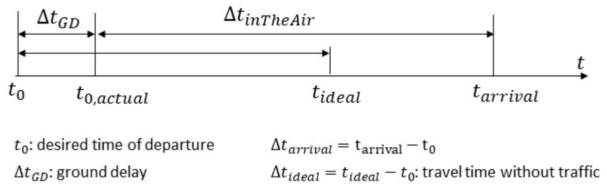
Δt_{GD}
$\Delta t_{inTheAir}$
t
t_0
$t_{0,actual}$
t_{ideal}
$t_{arrival}$
t_0: desired time of departure
Δt_{GD}: ground delay
$\Delta t_{arrival} = t_{arrival} - t_0$
$\Delta t_{ideal} = t_{ideal} - t_0$: travel time without traffic

Fig. 2 Definition of the time intervals of interest.

Figure 2 introduces the different time intervals that are used to define the efficiency metrics. $t_{0,actual}$ is the actual take-off time of the vehicle. Two different metrics are introduced to represent the time and efficiency metric.

The time efficiency is defined as:

$$\eta_{time} = \frac{\Delta t_{ideal}}{\Delta t_{arrival}} \tag{1}$$

The energy efficiency is defined as:

$$\eta_{energy} = \frac{\Delta t_{ideal}}{\Delta t_{intheair}} \tag{2}$$

Here time in flight is used as an approximation for the energy required for the flight. Time in flight was selected rather than path length due to the fact that hovering is part of some of the strategies being evaluated. In [9], the authors propose to use Work obtained by integrating the forward thrust along the path as the energy efficiency metric, but this requires a model of the vehicle and will not work well if vehicles can hover.

More efficient paths will lead to efficiency metrics values close to 1. As paths become less optimal, their efficiency metrics values will tend toward 0. The time efficiency is inferior or equal to the energy efficiency because $\Delta t_{arrival} > \Delta t_{inTheAir}$ by definition.

Agents are removed from the simulation when they are within a tolerance distance from their goal. To compute the efficiency the agent is considered to have traveled in a straight line at max speed from the point where they have been removed from the simulation to their goal.

2. Safety

In a traffic management context, safety can be defined in terms of loss of separation (LoS). Minimum separation distances define a protected volume around a vehicle (the ownship) in which other vehicles (the intruders) are not allowed to penetrate. A LoS happens when an intruder penetrates this protected volume. The number of LoS per flight hour, n_{LoS}/h, is one of the metrics used for measuring safety. However, using the number of LoS per flight hour as the only measure of safety is not sufficient. As explained in [19], a conflict severity metric should be used to express the degree to which a loss of separation was close to a collision. In concrete terms, when comparing two scenarios where the same LoS happens, the scenario where the LoS leads to a lower minimal distance between ownship and intruder should be penalized more. The Horizontal Intrusion Parameter (HIP) accounts for this. It is defined as:

$$HIP = 1 - \min_{t \in [t_{SOC}, t_{EOC}]} \left(\frac{\Delta s(t)}{S_{std}} \right) \tag{3}$$

Where t_{SOC} and t_{EOC} are the times at which the LoS starts and ends, $\Delta s(t)$ is the horizontal distance between the aircraft at time t, and S_{std} is the minimum horizontal separation distance required. A HIP value of 0 means that the two vehicles are at the appropriate horizontal distance. A HIP of 1 means that the two vehicles are at the same horizontal coordinates. This however does not necessarily mean that there is a collision since the vehicles could be at different altitudes (with the difference in altitude being less than the required vertical distance).

As will be seen in the results section, most LoS that occur in the alternatives that are studied have a low average severity. To better capture the likelihood of a severe LoS, the number of Near Mid-Air Collisions (NMAC) is also measured. NMAC occur when two aircraft get within 500 feet of each other [23]. Here, the number of NMAC per flight hours, n_{NMAC}/h is used to measure how common severe intrusions are in the different alternatives.

3. Capacity

The traffic volume or density alone is not sufficient to evaluate the capacity of an airspace. Similarly to what is done when analyzing car traffic, the throughput, i.e. the number of aircraft exiting the airspace per minute, should also be evaluated [15]. In [24], the authors propose to use practical capacity to measure the scalability of airspace systems. Practical capacity is defined as the number of aircraft that the airspace can accommodate before the average delay becomes greater than a certain threshold. In the proposed framework, the evolution of throughput and delays in function of agents density will be evaluated.

Throughput is related to the number of agents in the simulation, the average ideal travel time and the time efficiency. Trends in time efficiency will be reflected in the throughput, all other things being equal.

B. Simulation Approach

A simulation which allows the generation and evaluation of multiple alternatives was developed. The simulation does not implement all the methods that have been proposed in the literature. The following subsystems were implemented:

- Airspace structures: Free, Layers (with customizable range).
- Access Control: None, 4DT contract.
- Preflight Planning: Decoupled, SIPP, Local collision avoidance maneuver with a novel implementation
- Collision Avoidance: MVP, ORCA

The next subsections go into more details into the implementation and present the algorithms used for preflight planning and collision avoidance.

1. Overview

A custom 2D simulation framework implemented in Python was developed [*]. This agent-based simulation allows the generation and evaluation of alternatives in the same framework. Users can set the airspace structure, access control, preflight planning, and collision avoidance used by the agents.

Two distinct types of agents are modeled in the simulation: reactive or strategic. Reactive agents use a decentralized strategy to avoid conflict. At every time step these agents retrieve the position and velocity of neighboring agents and compute their next action. Strategic agents must submit a conflict-free 4DT flight plan to a centralized manager before starting their flight. The centralized manager stores all flight plans.

The simulation is synchronous, i.e. each agent is updated at each time step. A time step is made up of the following stages:

1) Each agent computes its next move based on its type.
2) Each agent's position gets updated.
3) Each agent close enough to its goal is removed from the simulation.
4) A conflict check is performed.

During a time step, the simulation iterates twice (stages 1 and 2) through all the agents: separating the agent decision-making and acting in two separate loops ensures that the order in which the decisions are made does not impact the solution. The trajectory of all agents in the simulation is represented as a series of segments, along which the agent has a constant speed. In this preliminary framework kinematics constraints are not being considered, e.g. the agent can change heading and velocity instantaneously, but its maximum velocity is bounded. In reality, vehicles will have limited acceleration and turn radius which will be highly dependent on the vehicle configuration. Here the assumption is that the vehicles have fast dynamics and the error between the desired trajectory and the actual trajectory would be negligible. Adding realistic kinematics and looking at the impact on the architecture's performance would be a different study.

To study one architecture at a constant density of agents in the airspace, the following procedure is used. The airspace starts completely empty. In the first phase agents are added to the simulation environment at a constant rate while also compensating for all agents that exit the simulation. This means that the total number of agents in the simulation increases at a constant rate during this phase. Once the desired number of agents has been reached, which is denoted as time t_0, the simulation enters a second phase. In the second phase, a new agents is only added when another agent exits so that the number of agents in the simulation is kept constant. Once all the agents in the simulation have been created after time t_0, the simulation reaches phase 3. This last phase is considered to be a steady state for the simulation.

In the experiments presented in section V the density was varied between 10 and 300 agents. When evaluating an alternative at a given density each run was repeated 10 times. The run is stopped once 100 agents that were created after the start of phase 3, have reached their goals. All agents that exit the simulation after t_0 are considered when evaluating the metrics. The agents are assumed to have a maximum velocity of 20 m/s. The simulation area is 20 by 20 km. The minimum separation distance is 500 meters. For reactive agents it is important to have a quick update rate as agents plan at every time step and must react to other agents' decisions. For strategic agents since the planning is done off-line the update rate can be larger to speed up the simulation. The simulation is updated every second for reactive agents and every ten seconds for strategic agents.

2. Preflight Planning

Preflight planning methods require a centralized manager to coordinate flights. All vehicles to submit a LoS-free flight plan before taking-off. The approaches that are presented here are global planners for each agent, i.e. each agent optimizes their own trajectory. Access to the airspace is regulated on a first-come first-serve basis, which is appropriate for on-demand operations. Three different methods are proposed: a decoupled method, an alternative to A* named Safe Interval Path Planning and a method that optimized the path locally.

Decoupled The decoupled approach that was implemented in this framework works by first fixing the agent path to be a straight line from its start to its goal. The path is then discretized into segments with a length equal to the minimum separation distance. The agent speed can either be zero or full speed. From a segment, the agent can either travel to the next or previous segment which would take a time dt or hover in place for dt. This defines a graph of the trajectories the agent could take. The well-known A* algorithm is used to find the shortest path in that graph [25].

[*]available at: https://github.com/colineRamee/UAM_simulator_scitech2021 (MIT license)

Algorithm 1 A* Algorithm

```
priorityQueue.push(start,0)
while current!=goal do
    current=priorityQueue.pop()
    for neighbor in current.getNeighbors() do
        g=current.costToGo+travelCost(current,neighbor)
        h=heuristic(neighbor)
        f=g+h
        if not neighbor.open OR neighbor.costToGo>g then
            priorityQueue.push(neighbor,f)
            neighbor.open=True
            neighbor.costToGo=g
            neighbor.parent=current
```

Safe Interval Path Planning The Safe Interval Path Planning method was introduced in [26]. It is a variation of the A* algorithm that reduces the dimensionality of the time dimension by reasoning on time intervals instead of discretizing the time dimension in fixed increments like the A* implementation used above. The time intervals represent continuous time segment where a point on the grid is free of conflict. The algorithm assumes that it is always advantageous to move forward and that the vehicle can wait in place. The SIPP method maximizes the time efficiency.

The main difference between the A* algorithm and SIPP is a change at line 4 of the Algorithm 1. The getNeighbors() method is replaced by a getSuccessors method that query accessible intervals instead of neighboring grid cells. As can be seen in Algorithm 2, this method requires to find the first time at which the agent can arrive on the new interval. This requires to determine what is the earliest time at which the agent can start traveling toward the new interval without losing separation with other agents. Details on the equations used are available in Appendix II.

The simulation area is discretized as a grid. Each grid cell measures 500m by 500m. The grid is 8-connected, i.e. an agent can move from a grid cell to one of its eight neighbors (the four it shares a border with and the four on the diagonals). The centralized planner keeps track of agents flight plans and stores free intervals. To find which grid cells are covered by an agent's flight plan, a digital differential analyzer (DDA) line algorithm is used. Because agents can travel along diagonals a collision check is necessary in addition to checking for free intervals.

Algorithm 2 getSuccessors(current)

```
successors = []
for neighbor in current.getNeighbors() do
    delta_t = travelTime(current,neighbor)
    start = current.time +delta_t
    end = current.end +delta_t
    for interval in neighbor.getSafeIntervals() do
        if [start,end]$\cap$interval $\neq \emptyset$ then
            t = getFirstArrivalTime(current, interval)
            if t $\neq \emptyset$ then
                successors.append((interval, time))
return successors
```

Local Local collision avoidance methods can be used to plan a collision free path as done by [14]. Here we propose to use a Velocity Obstacle (VO) based approach. We propose to formulate the collision avoidance problem as a a Mixed Integer Quadratically Constrained Programming (MIQCP) problem. It is solved using the commercial solver Gurobi.

VO methods define sets of velocities for the ownship that would eventually result in a collision if the intruder were to maintain its current velocity. Collision avoidance methods based on VO then aim at finding a velocity outside of the union of all VOs. Methods to find a velocity outside of these collision velocity sets often use a heuristic. Typically methods based on VO either use sampling or computing the intersection of the sets to find feasible velocities [27]. In the ORCA algorithm which will be presented later, VO are simplified to be expressed as a single constraint, allowing the problem to be solved using a linear programming algorithm with a quadratic constraint on the norm [28].

In our approach finding a velocity outside the union of the VO set was formalized as a mixed-integer quadratically constrained program (MIQCP) problem. The ownship velocity is expressed as $V = [v_x, v_y]$.

The VO set can be defined as all v_x, v_y such that $s_{i,1}(v_x, v_y) \geq 0$ **and** $s_{i,2}(v_x, v_y) \geq 0$, where $s_{i,1}(x, y) = 0$ and $s_{i,2}(x, y) = 0$ are the equations of the lines forming the VO cone generated by intruder i, as illustrated in Figure 3.

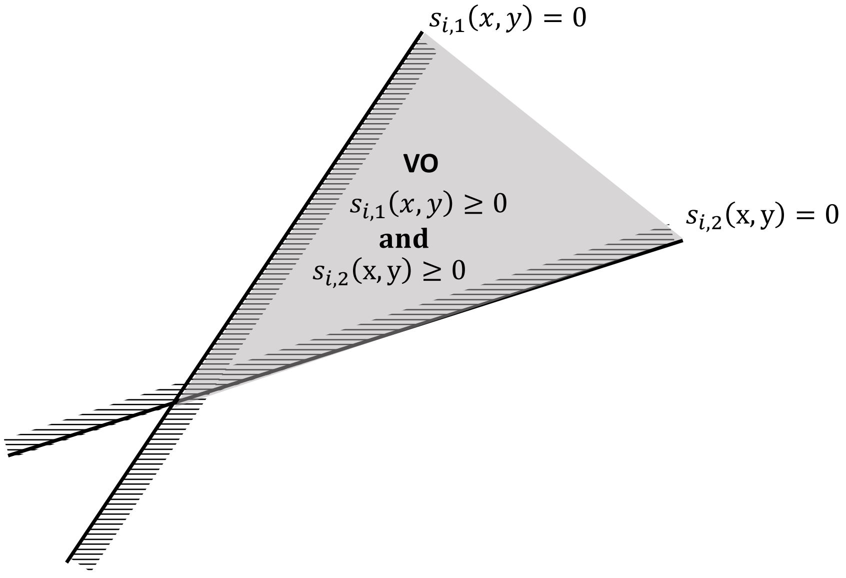

Fig. 3 Expressing the Velocity Obstacle as two linear constraints

In order to get velocities outside the VO, the complement of that set must be obtained. The constraints from an intruder i can be expressed as:

$$\begin{aligned} s_{i,1}(v_x, v_y) &\leq 0 \\ &OR \\ s_{i,2}(v_x, v_y) &\leq 0 \end{aligned}$$

To represent the OR constraint in a linear framework, integer variables are introduced similarly to what is done in [29].

$$\begin{aligned} s_{i,1}(v_x, v_y) - K * a_{i,1} &\leq 0 \\ s_{i,2}(v_x, v_y) - K * a_{i,2} &\leq 0 \\ a_{i,1} + a_{i,2} &\leq 1 \end{aligned}$$

Where $a_{i,1}$ and $a_{i,2}$ are binary variables. K can be chosen to be arbitrarily large. However selecting a value too large for K can create issues for the solver, making some elements poorly conditioned and creating issues with relative tolerance values. Hence for a practical implementation the value of K is selected based on the maximum value of $s_{i,1}$ and $s_{i,2}$ for v_x and v_y in the range $[-v_{max}, v_{max}]$. A final quadratic constraint comes from defining a maximum velocity for the agent. This is the only constraint that is quadratic.

$$v_x^2 + v_y^2 \leq v_{max}^2$$

Finally, the objective of the optimization problem is to minimize the difference between the chosen velocity and the desired velocity. The desired velocity is simply a vector pointing towards the goal that has an intensity of v_{max}, this is the velocity that would be optimal if there were no intruders.

$$\underset{v_x, v_y}{\text{minimize}} (v_{desired,x} - v_x)^2 + (v_{desired,y} - v_y)^2$$

This optimization problem is solved using the Gurobi software python interface [30].

Algorithm 3 illustrates how the velocity obstacle solver is used in a centralized strategic method. First, the algorithm finds the first suitable take-off time that occurs after the agent's desired time of departure. This is performed by checking that there is no agent within the minimum separation of departure at the start time and if there is adding ten seconds to the start time and continuing to check until a solution is found. Once a suitable start time has been found, the agent's trajectory is constructed by solving a MIQCP problem every ten seconds. If at some point the solver cannot find a solution then it means that there is no way to avoid the conflict given the state of the trajectory. The trajectory is reset and the start time used to generate this trajectory is incremented. This gets repeated until a valid trajectory is found. A valid trajectory will always exist due to the first come first serve priority scheme: if the agents wait long enough there will be no planned aircraft in the air.

The ownship velocity is updated every 10 seconds. Intruders that are being considered are limited to the 10 closest intruders within a radius of 5 km from the ownship.

Algorithm 3 Local VO

```
while NOT FoundSafeTrajectory do
    startTime = manager.FirstAvailableTakeOffTime(startTime, startPosition)
    time=startTime, position=startPosition, trajectory=[(time,position)]
    while position ≠ goal AND NOT noSolution do
        intruders = manager.getPlannedIntruders(position, time)
        desiredVelocity = getOptimalVelocity(position, goal, maxSpeed)
        model = setupMIQCP(intruders, position, desiredVelocity)
        model.solve()
        if model.hasSolution() then
            velocity = model.solution
            position += velocity * dt
            time += dt
            trajectory.append((time,position))
        else
            noSolution = True
    if noSolution then
        startTime +=dt
    else
        FoundSafeTrajectory = True
return trajectory
```

3. Collision Avoidance

MVP A reactive decentralized strategy called Modified Voltage Potential (MVP) was implemented. The MVP method is a geometric method initially developed by Eby in [31], which was then improved by Hoekstra as illustrated in [32] using the Velocity Obstacles concept.

A difference between MVP and other VO method is that it only considers intruders that are in direct conflict with the ownship instead of all neighboring intruders. MVP works by finding the time of closest approach between the ownship and an intruder. A derivation of the formula for the time of closest approach is provided in Appendix I. If there is a conflict, i.e. if the distance between the intruder and the ownship is projected to be less than the minimum separation distance, then the algorithm computes the intrusion vector and identifies the velocity change required for the ownship to avoid the conflict. In cases where there are multiple conflicts, the velocity changes required are summed. The method provides an exact solution when there is only one intruder and its velocity is constant, and yields a usually satisfying approximation otherwise.

The implementation in this framework is based on the implementation provided in the Bluesky simulator [†]. A 10% safety factor is added to the minimum separation distance to compensate for floating point errors and reduce the number of LoS.

[†]https://github.com/bluesky/bluesky

ORCA As pointed out in [33], one issue that arises when all agents are using a VO-based method for collision avoidance at the same time is the potential development of oscillations. This can also occur with MVP. The ORCA method was developed to address the issues that occur with collaborative VO-methods.

A brief overview of the ORCA algorithm is given here, for a detailed explanation the reader is directed to the paper by Berg et al. that presents the algorithm [28]. ORCA formulates the VO sets slightly differently than what was presented in the local VO method. It adds a time horizon, meaning that conflict that would arise after a certain time should not be considered. This time horizon manifests itself by rounding the tip of the VO cone. ORCA simplifies the problem of finding the optimal velocity outside of the union of the VO sets by representing the constraint created by an intruder as a single constraint tangent to the VO created by that intruder. All agents that use ORCA rely on the fact that other reactive agents also use that same algorithm and that the avoidance burden can be shared in half.

A C++ implementation of ORCA is available open-source in the RVO2 library [‡]. The code was adapted for python to work in the proposed framework. ORCA requires some parameters to be chosen by the user. The range, i.e. the maximum distance at which intruders are being considered, was set to 5km similarly to the local VO method. The time horizon was set to be equal to the range divided by the max velocity of the agents which yielded 250 seconds or 4.2 minutes. Following examples of the ORCA method and indications given by the authors in the paper and code, the maximum number of intruders that are considered was set to 10.

In [34], the authors apply the ORCA algorithm to aircraft with velocity constraints. They show that for aircraft entering conflicts with a shallow difference in heading, the ORCA algorithm tends to increase the length of the conflict by making aircraft take parallel tracks rather than resolving the conflict.

4. Airspace Structure

Two types of airspace structure were considered here, free airspace and layers.

Layers segregate vehicles to a specific altitude based on the heading required to go from their origin to their destination. Vehicles stay at the same altitude throughout their flight, which is why a simple 2D approach can be used to model it. To restrict the range of headings in the simulation, a simple check is used when generating the start and goal pair for one agent. If the pair falls in the heading range being considered then that pair is kept. If the heading from start to goal is outside the range, then a new start and goal pair is redrawn until a satisfying pair is found. Since the demand is supposed to be uniform, a single layer can be modeled.

The different ranges of heading that are considered here are $[0°, 180°]$, $[0°, 120°]$ and $[0°, 90°]$, which respectively correspond to a minimum of 2, 3 and 4 layers to cover all possible flights.

V. Results

In this section multiple alternatives are evaluated using the simulation presented in section IV.B.1. However, it is important to keep in mind that a number of simplifying assumptions were made when building the simulation. Vehicle dynamics, approach and departure procedures, external factors, such as weather or non-participating traffic, were all ignored. The simulation is limited to 2D, and after submitting a 4DT contract agents are assumed to follow the contract perfectly. Still, this simplified framework can give good insight on the potential limitations of some alternatives.

The first study looks at how the different free airspace alternatives perform with an increasing number of agents. The second study compare these free airspace alternatives to layered alternatives and vary the range of layers. A baseline in which agents simply travel in straight line between their origin and destination without avoiding each other is compared to the proposed alternatives. Table 2 lists all the alternatives that are compared in this section.

A. Performance of different access control, preflight planning and collision avoidance methods in a free airspace

As shown in Figure 4, the throughput increases as the number of agents in the simulation is increased, but the rate at which it increases is very different between alternatives. The baseline shows the maximum throughput that could be attained if agents could travel in a straight line without slowing down and avoiding each other. The reactive MVP method and the strategic SIPP and Local VO methods show similar performance in terms of throughput, with SIPP taking an advantage at very high density. The ORCA method accommodates a smaller throughput than these three methods with the difference becoming larger as the agent density increases. The decoupled method has a much more limited throughput than the other methods and appears to tend toward a limit of 10 agents/min.

[‡]http://gamma.cs.unc.edu/RVO2/

Table 2 List of Alternatives Studied in the Results Section

Airspace Structure	Access Control	Preflight Planning	Collision Avoidance
Free	None	None	Reactive, MVP
Free	None	None	Reactive, ORCA
Free	None	None	None
Free	4DT contract	Strategic, SIPP	None
Free	4DT contract	Strategic, Decoupled	None
Free	4DT contract	Strategic, Local VO	None
Layers	None	None	Reactive, MVP
Layers	None	None	Reactive, ORCA
Layers	None	None	None
Layers	4DT contract	Strategic, SIPP	None
Layers	4DT contract	Strategic, Decoupled	None
Layers	4DT contract	Strategic, Local VO	None

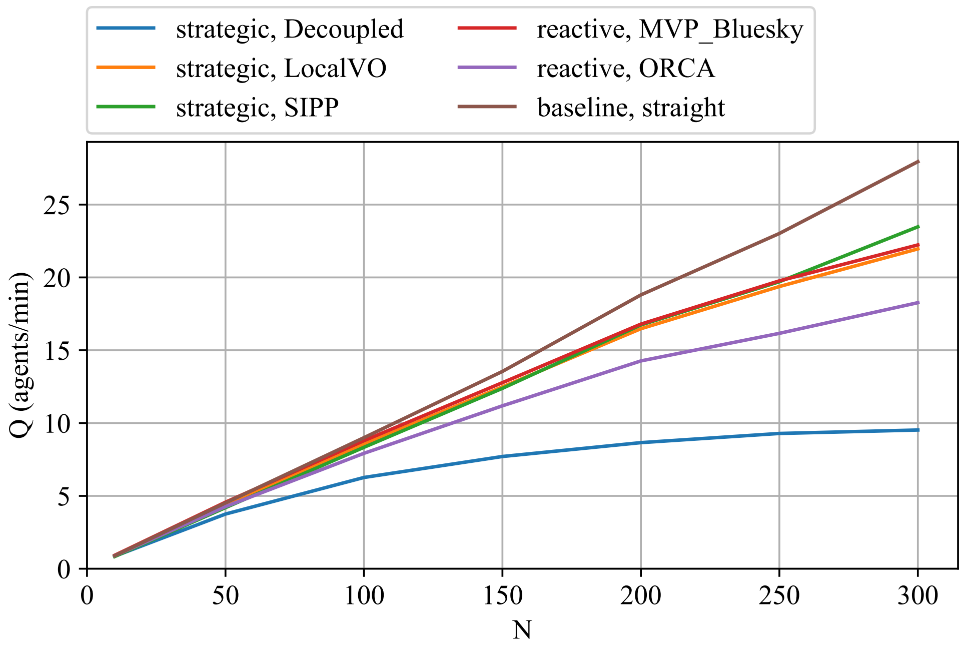

Fig. 4 Evolution of throughput Q in function of agents density N for different alternatives with a free airspace structure

Figure 5 gives some insight as to why the decoupled method performs so poorly. Its energy efficiency is somewhat lower than other methods, meaning that it waits in the air for agents to pass which is less efficient than performing an avoidance maneuver in terms of time it takes to reach the goal, but it is the time efficiency that is significantly lower than other methods. This means that the decoupled method imposes long ground delay to its agents. The SIPP method has a lower energy and time efficiency than other methods at low density. This is because it constrains agents movements to be on an 8-connected grid, meaning that agents can only turn by 45 degrees. For instance, if the origin is at position $(0, 0)$ and the goal is at $(10, 20)$, the angle between the origin and the goal is 30 degrees and the agent shortest path on the grid is 8% larger than the straight line distance. As the density of agents in the simulation is increased the benefits of the SIPP algorithm becomes more apparent. Starting at 250 agents it performs better than the Local VO method, although

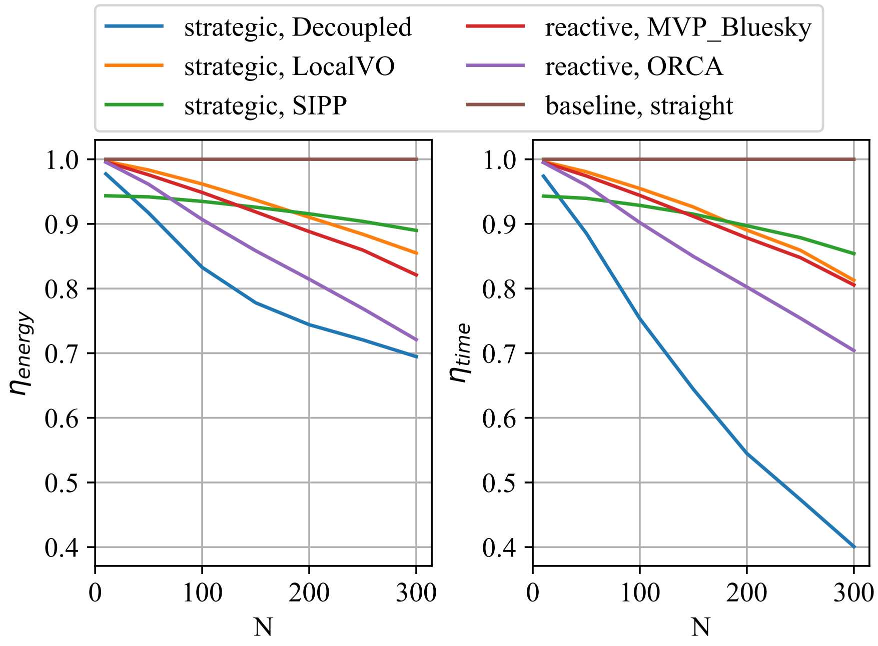

Fig. 5 Evolution of energy (left) and time (right) efficiencies in function of agents density N for different alternatives with a free airspace structure

the difference remains small. Comparing energy and time efficiency shows that reactive method time efficiency is closer to their energy efficiency. This is because they only experience ground delay if there are intruders within the minimum separation distance from their origin, while strategic method plan ground delays to ensure a LoS free flight, which can result in longer delays.

As can be seen in Figure 6, in the baseline case, the number of conflict increases with the number of agents. An increase in the number of conflict means the agents have to perform more maneuvers to avoid LoS, which reduces the energy efficiency of the agent. Although the agents decision is likely to impact the number of conflict, this helps understand why efficiency decreases with the number of agents for all alternatives.

By definition, the centralized strategic reactive alternatives do not have any loss of separation. Figure 6 shows that MVP fails more often to maintain the minimum separation distance between intruders than ORCA, and that the number of LoS per flight hour increases with the density of agents in the simulation. However, looking at the average HIP shows that for both ORCA and MVP the value remains low and roughly constant with the number of agents in the simulation. This suggests that most LoS have a low severity, and indeed looking at the distribution of HIP on Figure 7 shows that three quarters of LoS for both reactive methods have a HIP value lower than 0.2, meaning that agents remain at least 400 meters apart. Although the number of LoS per flight hour is much higher for the MVP architecture than for the ORCA architecture, the number of NMAC per flight hour is roughly similar between the two. Depending on how much value is placed on avoiding non-severe LoS versus improving the throughput, a decision maker might choose a different reactive architecture.

A remark can be made for the baseline that the HIP is uniformly distributed, which results in an average value of 0.5. This might seem counter-intuitive at first but is easily explained when one remembers that there is a minimum distance term in the HIP expression. The probability that two randomly distributed points are at a distance from each other between $R - \frac{dr}{2}$ and $R + \frac{dr}{2}$ is proportional to $2\pi R dr$, i.e. agents are more likely to be far than close. To think about how minimum distance is different let's consider a fixed ownship and the set of all intruders that have parallel tracks, the minimum distance between the ownship and the intruders occur at the intersection between the intruder track and a line perpendicular to the intruder's track that goes through the ownship's position (see Appendix I). Since agents'start positions are uniformly distributed, the minimum distance position for the set of parallel intruders is uniformly distributed as well.

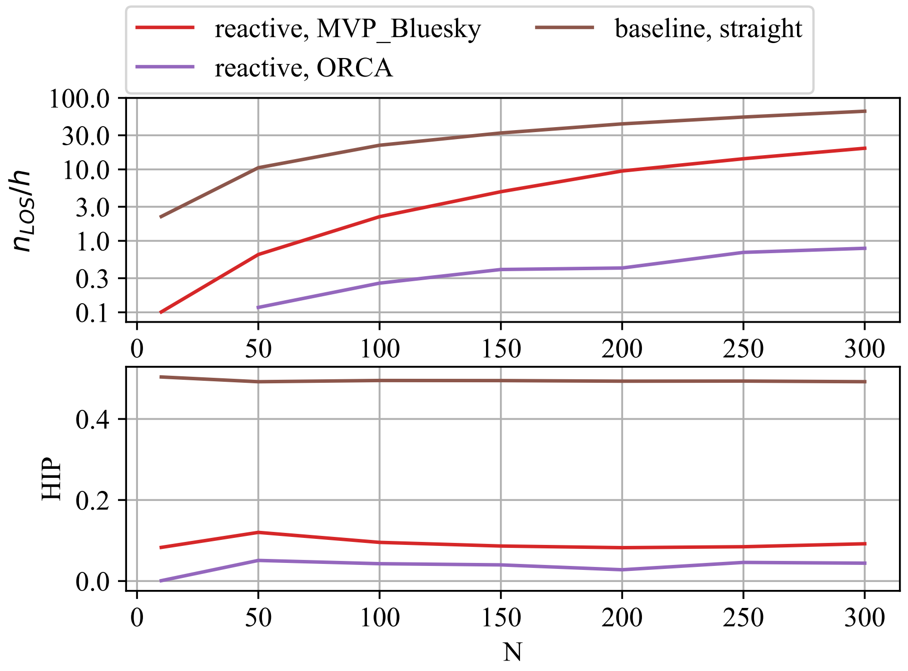

Fig. 6 Evolution of average number of losses of separation per agent flight hour (top, shown on a log scale) and HIP (bottom) in function of agents density N for different reactive alternatives

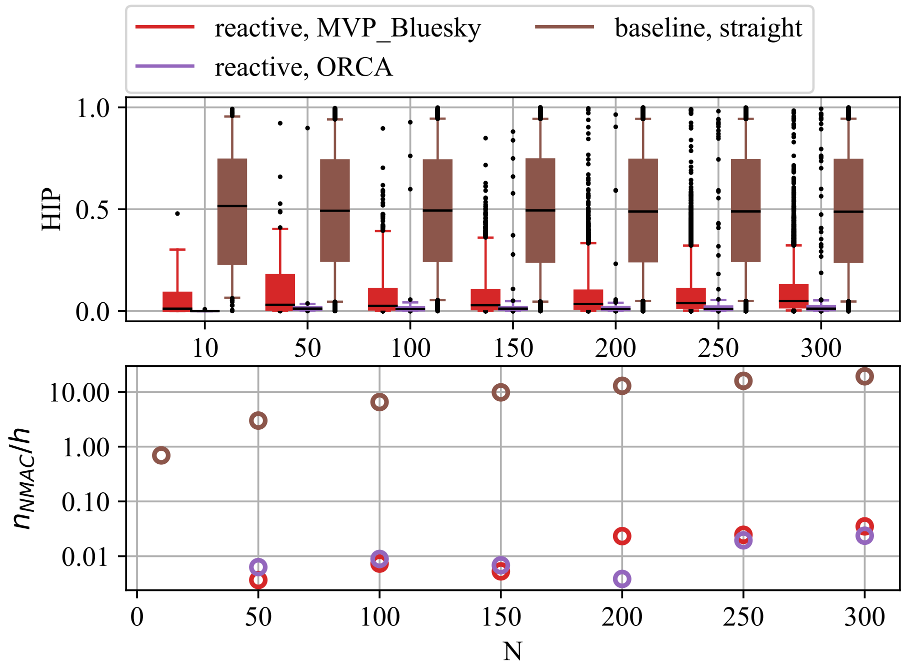

Fig. 7 Distribution of loss of separation severity as measured by the Horizontal Intrusion Parameter (top) and number of NMAC per agent flight hour shown in log scale (bottom) in function of agents density N for different reactive alternatives. The whiskers of the bounding boxes indicate the 5th and 95th percentiles.

B. Impact of airspace structure

Since agents do not change layers in flight in any of the alternatives being considered in this thesis, a simple 2D approach was used. To restrict the range of headings, a simple check is used when generating the start and goal pair for

one agent. If the pair falls in the heading range being considered then that pair is kept. If the heading from start to goal is outside the range, then a new start and goal pair is redrawn until a satisfying pair is found.
The different ranges of heading considered were $[0, 180]$, $[0, 120]$ and $[0, 90]$, which respectively correspond to a minimum of 2, 3 and 4 layers to cover all possible flights.

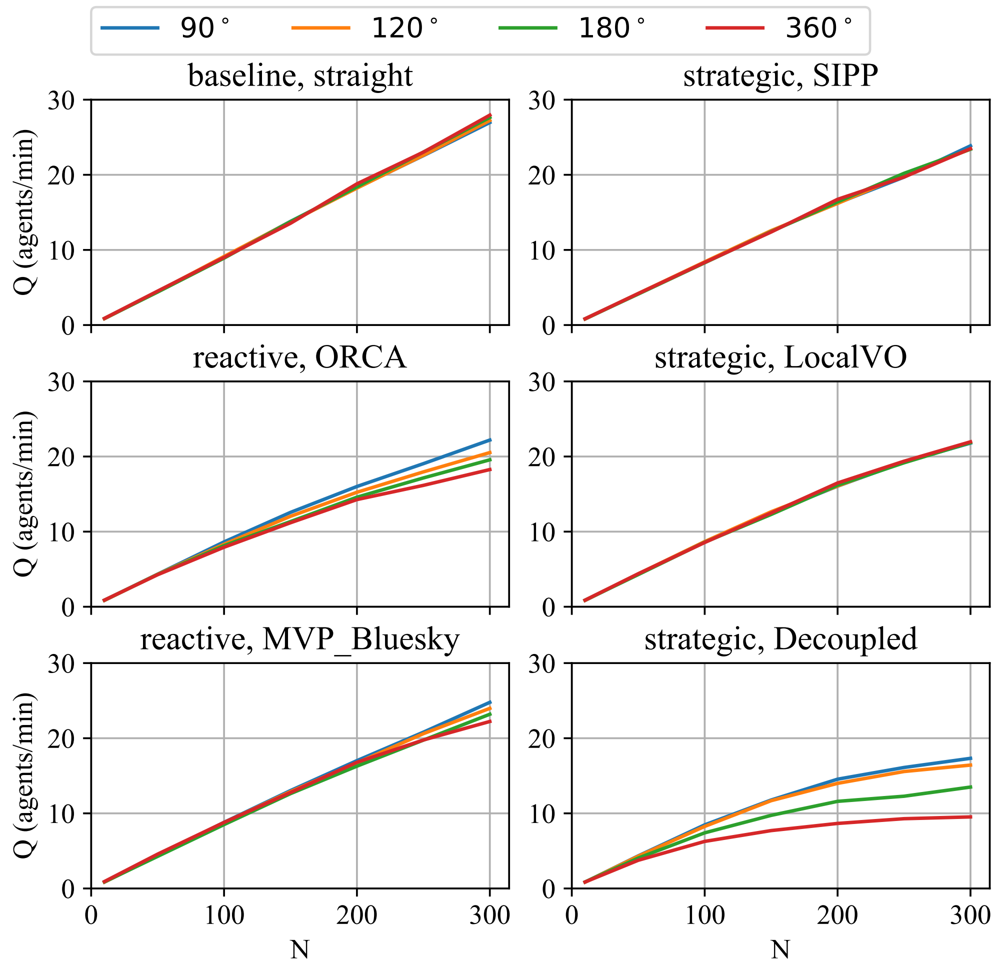

Fig. 8 Evolution of throughput Q in function of agents density N for different deconfliction strategies and layer ranges.

Analysis of the impact of segregating traffic by heading range for different deconfliction strategies shows that the impact is very different depending on the strategy. As can be seen on Figure 8, segregating the traffic by layer has very little impact on the baseline throughput, i.e. the average distance for origin/destination pairs remains roughly constant no matter the heading restriction. Among centralized strategic methods, only the decoupled approach shows that segregating the traffic by heading improves throughput performance, the local VO and SIPP algorithms have relatively constant throughput in function of heading range. For the decoupled method, the improvement added by each new layer diminishes, and although the throughput at 300 agents almost double when adding a 90° heading range restriction compared to the 360° baseline it is not enough to make the decoupled method competitive compared to other strategic methods at high density. The two reactive methods show an improvement as the heading range is reduced but less than the strategic decoupled methods.

Looking at reactive strategies on Figure 9 shows that layers have a more distinct impact on energy efficiency than on time efficiency. The only difference between the two efficiencies for reactive strategies is the time spent waiting on the ground for the airspace above the agent's origin to be clear from intruders. When the range of heading is restricted, the localization of the agent's origin also becomes constrained. If the direction between the destination and the origin

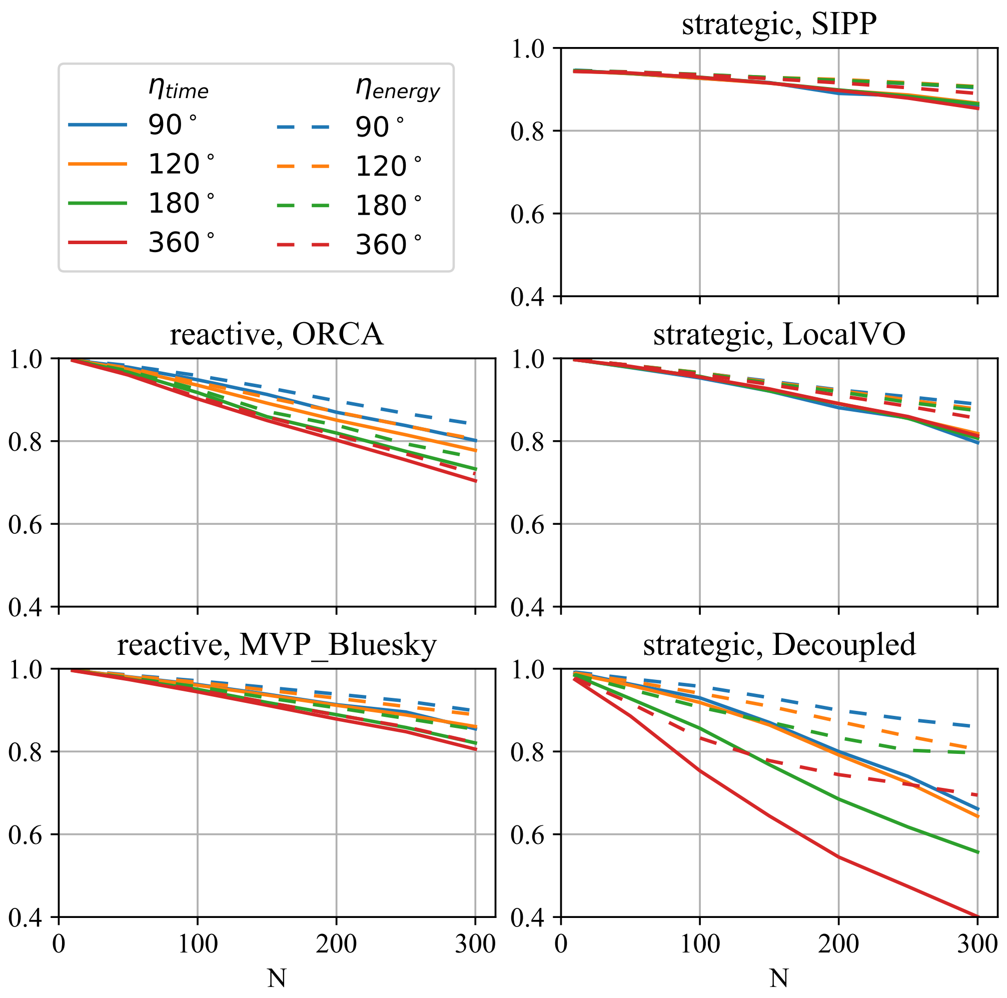

Fig. 9 Evolution of energy and time efficiency in function of agents density N for different deconfliction strategies and layer ranges.

must be between 0 and 90°, then the origin cannot be on the southern limit of the simulation area. This causes the density at the edge of the simulation area to increase since the same number of agents must be added and leads to higher wait times on the ground as the range of heading is reduced. This effect is linked to the simulation. Looking at energy efficiency shows that adding layers improve the average efficiency as could have been expected. For centralized strategic algorithms strategies, similarly to what was observed for throughput, the impact of layers is particularly clear for the decoupled strategy, but less so for the SIPP and local VO methods.

As can be seen on Figure 10, reducing the heading range decreases the number of losses of separation per flight hour for the baseline case. In the baseline case agents do not avoid each other and simply fly straight from their origin to their destination. This can give an indication of the number of conflicts that agents can expect to have to solve and the overall complexity of the airspace. This is imperfect since the algorithms used will change the behavior of agents and impact the number of conflicts, but still shows that as layers are added and the heading range decreases agents can expect to have to solve fewer conflicts. The more conflicts must be solved, the more agents will have to maneuver and the more the energy efficiency will be reduced. This is coherent with what was observed in Figure 9.

Figure 11 shows that in a system controlled by the reactive methods ORCA or MVP, the restriction of heading range does help to reduce the number of losses of separation per flight hour. In the ORCA case due to the relatively small number of losses of separation, the curve are somewhat noisy. The trends in term of severe loss of separation likelihood are opposite, with n_{NMAC}/h being higher the more the heading range is restricted. This might be due to the fact that the more the heading is restricted the smaller the perimeter of the simulation area where agents might be spawned is.

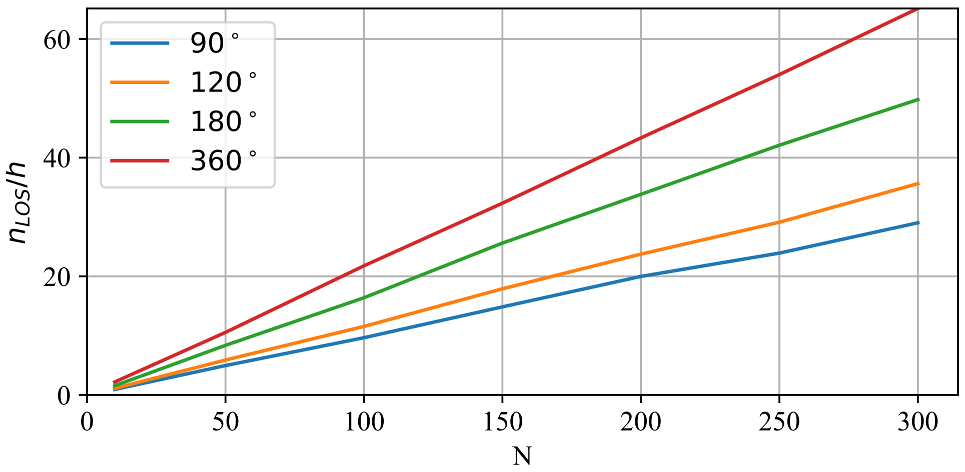

Fig. 10 Evolution of the average loss of separation per flight hour in function of agents density N for different layer ranges for the baseline straight strategy.

Indeed, consider an extreme case where agents are restricted to go east, then agents can only be spawned from the west when studying this layer. Hence agents are created with less distance from each other which might result in higher local densities.

VI. Conclusion

This work presents a system decomposition for low-altitude unmanned air traffic management systems into four subsystems: (1) airspace structure, (2) access control, (3) preflight planning, and (4) collision avoidance. It builds on the existing literature to suggest a set of metrics that can be used to compare UTM alternatives. A 2D simulation environment that can be used to evaluate alternatives is introduced. An overview of the implementation is given and details about the subsystem alternatives currently implemented in the simulation are provided. A novel local centralized planning method and an adaptation of a method from the robotics community to a UTM system are presented. This framework consisting of the metrics and simulation environment is showcased by comparing how different deconfliction strategies perform in a free airspace, and then studying how variations in the airspace structure affect these strategies.

This shows that although reactive methods have large number of losses of separation at high densities, the number of severe losses of separation or NMAC was much lower. A local collision avoidance method was shown to perform comparably to a grid optimal path planning method and much better than a decoupled approach at high densities, indicating that to simplify the 4D trajectory problem keeping flexibility in the spatial trajectory was important. Indeed,the decoupled approach to trajectory planning was shown to perform poorly and is not scalable at high densities, even when traffic is segregated by heading. The impact of segregating traffic by heading was very different depending on the alternative, which shows that it is important to consider airspace structure at an early stage of the decision-making process.

Although this work makes a number of simplifying assumptions, it shows some interesting results and lays the foundation for future developments. Future work will be conducted to evaluate the impact of external factors such as non-participating traffic, the presence of static obstacles in the simulation area and non-uniform traffic demand.

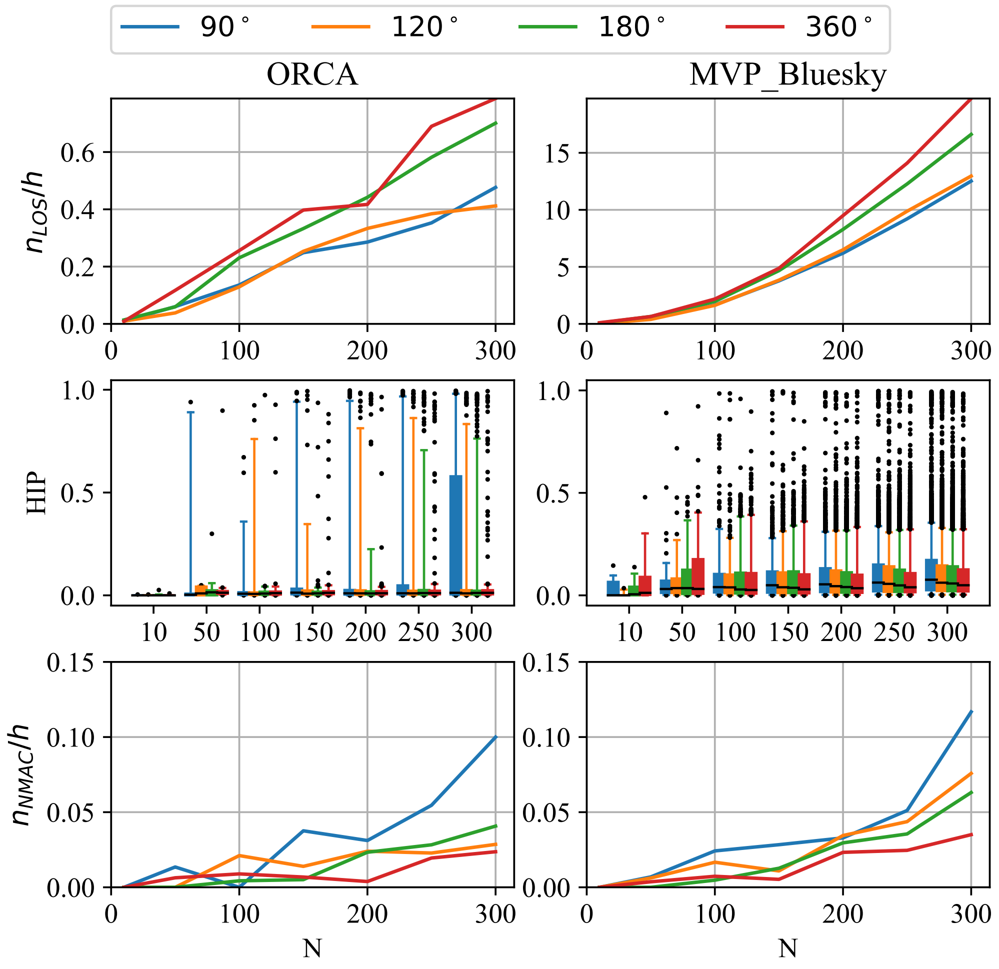

Fig. 11 Evolution of the average loss of separation per flight hour (top), distribution of Horizontal Intrusion Parameter (middle), and average near mid-air collision per flight hour (bottom) in function of agents density N for different layer ranges for the ORCA (left) and MVP (right) algorithms.

Appendix 1: Time Of Closest Approach

Let's consider two agents A and B, whose positions are respectively $\mathbf{P_A}(t_0)$ and $\mathbf{P_B}(t_0)$ at time t_0, and velocities are the constant $\mathbf{V_A}$ and $\mathbf{V_B}$. We want to find the time t^* at which the distance between the two agents is minimum.

We define $\mathbf{R}_{AB}(t)$ the relative position of agent B with respect to A at time t, and write the distance between the two agents at time t as $r_{AB}(t)$. By definition:

$$\begin{aligned}
\mathbf{R}_{AB}(t) &= \mathbf{P_B}(t_0) + (t - t_0) * \mathbf{V_B} - \mathbf{P_A}(t_0) - (t - t_0) * \mathbf{V_A} \\
r_{AB}(t) &= ||\mathbf{R}_{AB}(t)|| \\
r_{AB}(t) &= \sqrt{\mathbf{R}_{AB}^T.\mathbf{R}_{AB}}
\end{aligned}$$

Solving for $\min_t(r_{AB})$ is equivalent to solving for $2\frac{d\vec{R}_{AB}}{dt}^T.\vec{R}_{AB} = 0$ and hence:

$$t^* = t_0 - \frac{(\mathbf{V_B} - \mathbf{V_A})^T.(\mathbf{P_B}(t_0) - \mathbf{P_A}(t_0))}{||\mathbf{V_B} - \mathbf{V_A}||^2} \tag{4}$$

If $\mathbf{V_A} = \mathbf{V_B}$ the distance between the agents is constant and t^* is set equal to t_0.

As illustrated on Figure 12 the distance of closest approach can be found graphically and equation 4 interpreted as the projection of the relative position on the relative velocity vector. To interpret this figure, similarly to what is done when building a Velocity Obstacle set, we reason in the frame of the intruder, i.e. the velocity of the intruder is subtracted to the ownship and the intruder's position is considered constant.

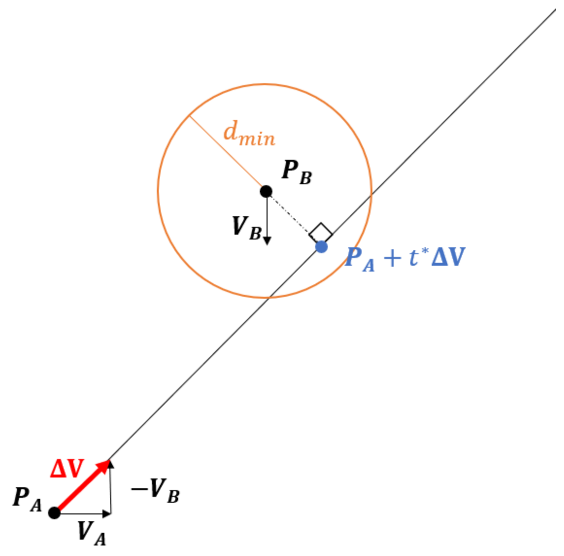

Fig. 12 Finding the distance of closest approach by reasoning geometrically in the intruder reference frame

Appendix 2: Time to Leave

Assuming agent A and agent B will lose separation if agent A leaves its current position at a velocity $\mathbf{V_A} \neq 0$, we want to find a delay t' if it exists for which the minimum distance between the two agents is exactly the minimum separation distance.

$$r_{AB}(t^*) = d_{min}$$

The position of agent A at time t can be expressed as:

$$\mathbf{P_A}(t) = \begin{cases} \mathbf{P_A}(t_0) + (t - t')\mathbf{V_A}, & \text{if } t > t' \\ \mathbf{P_A}(t_0), & \text{otherwise} \end{cases}$$

Let's define $\Delta\mathbf{P} = \mathbf{P_B}(t_0) - \mathbf{P_A}(t_0)$ and $\Delta\mathbf{V} = \mathbf{V_B} - \mathbf{V_A}$
Assuming $r_{AB}(t^*(t' = 0)) < d_{min}$ and $t' < t^*$, the formula for t^* as a function of t' can be derived similarly to what was done in the previous section.

$$r_{AB}(t, t') = ||\Delta\mathbf{P} + \Delta\mathbf{V} \cdot t + \mathbf{V_A} \cdot t'|| \tag{5}$$

$$t^*(t') = t_0 - \frac{\Delta\mathbf{V} \bullet (\Delta\mathbf{P} + \mathbf{V_A} \cdot t')}{||\Delta\mathbf{V}||^2} \tag{6}$$

We can plug back equation 6 into equation 5 to find the minimum distance between the agents as a function of the delay.

$$r_{AB,min}(t') = r_{AB}(t^*(t'))$$

$$r_{AB,min}(t') = \left\| \Delta\mathbf{P} + \Delta\mathbf{V} \cdot (t_0 - \frac{\Delta\mathbf{V} \bullet (\Delta\mathbf{P} + \mathbf{V_A} \cdot t')}{||\Delta\mathbf{V}||^2}) + \mathbf{V_A} \cdot t' \right\| \tag{7}$$

We want to find t' such that $r_{AB,min}(t') = d_{min}$, since $d_{min} > 0$ and $r_{AB,min} \geq 0$ this is equivalent to solving the quadratic equation in t': $r_{AB,min}(t')^2 = d_{min}^2$.
To simplify the expression we introduce the following terms:

$$\mathbf{\Gamma} = \Delta\mathbf{P} + \Delta\mathbf{V} \cdot (t_0 - \frac{\Delta\mathbf{V} \bullet \Delta\mathbf{P}}{||\Delta\mathbf{V}||^2})$$

$$\mathbf{\Lambda} = \Delta\mathbf{V} \cdot (-\frac{\Delta\mathbf{V} \bullet \mathbf{V_A}}{||\Delta\mathbf{V}||^2}) + \mathbf{V_A}$$

Hence equation 7 becomes:

$$r_{AB,min}(t') = ||\mathbf{\Gamma} + \mathbf{\Lambda} \cdot t'||$$

And so the quadratic equation to be solved is:

$$d_{min}^2 = ||\mathbf{\Gamma}||^2 + 2\mathbf{\Lambda} \bullet \mathbf{\Gamma} \cdot t' + ||\mathbf{\Lambda}||^2 \, t'^2$$

The determinant of the quadratic equation is:

$$\Delta = (\mathbf{\Lambda} \bullet \mathbf{\Gamma})^2 - (||\mathbf{\Gamma}||^2 - d_{min}^2) \cdot ||\mathbf{\Lambda}||^2$$

If $\Delta \geq 0$ the possible delays are:

$$t'_a = \frac{-\mathbf{\Lambda} \bullet \mathbf{\Gamma} - \sqrt{\Delta}}{\mathbf{\Lambda}}$$

$$t'_b = \frac{-\mathbf{\Lambda} \bullet \mathbf{\Gamma} + \sqrt{\Delta}}{\mathbf{\Lambda}}$$

This can be interpreted geometrically similarly to what was done in the previous section. This helps understand the different domains of the solution. The minimum distance between the agents can either occur while agent A is moving

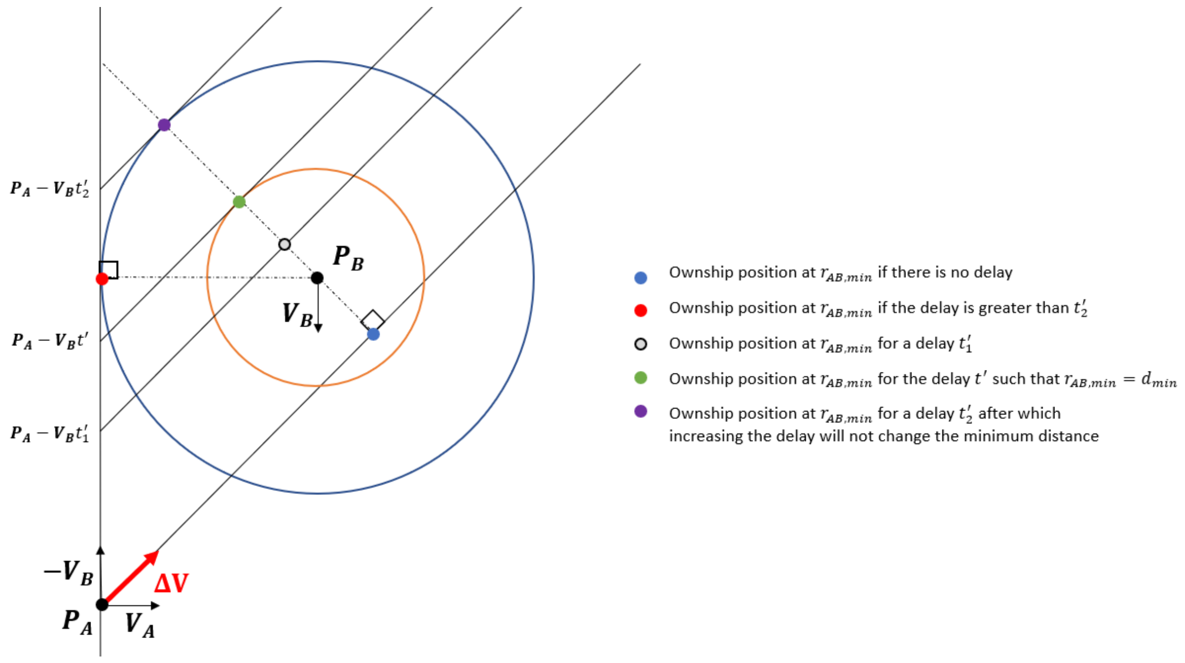
$P_A - V_B t_2'$
$P_A - V_B t'$
$P_A - V_B t_1'$
P_B
V_B
$-V_B$
ΔV
P_A
V_A
● Ownship position at $r_{AB,min}$ if there is no delay
● Ownship position at $r_{AB,min}$ if the delay is greater than t_2'
○ Ownship position at $r_{AB,min}$ for a delay t_1'
● Ownship position at $r_{AB,min}$ for the delay t' such that $r_{AB,min} = d_{min}$
● Ownship position at $r_{AB,min}$ for a delay t_2' after which increasing the delay will not change the minimum distance

Fig. 13 Finding the distance of closest approach for different delays by reasoning geometrically in the intruder reference frame

or while agent A is waiting. Figure 13 shows how the position at which the minimum distance occurs changes with the delay. Since the figure is shown in agent B reference frame, when agent A is waiting it moves along $-\mathbf{V_B}$, when agent A's delay has passed it moves along $\Delta\mathbf{V}$. So to find the time to leave on the figure simply find the point where $\Delta\mathbf{V}$ is tangent to the minimum separation circle, the intersection between the tangent and the waiting line $-\mathbf{V_B}$ gives the position where agent A has to leave $\mathbf{P_A} - \mathbf{V_B}t'$, where t' is the time to leave. The blue circle shows the minimum distance if the ownship waits indefinitely $d_{min,wait}$. The two agents are at this distance at time $t_{min,wait}$. As can be seen on the figure, if agent A leaves at that time or shortly after, the minimum distance does not occur at $t_{min,wait}$ but on the segment where agent A is moving. The purple dot and its associated delay t_2' shows when the minimum distance when not moving is equal to the minimum distance while moving with a delay. For any delay $t <= t_2'$ the minimum distance occurs when the agent is moving, for any delay $t > t_2'$ the minimum distance occurs at $t_{min,wait}$ when the agent is waiting.

As shown on Figure 14 if $d_{min,wait}$ is smaller than the desired separation distance d_{min} then there is no solution for the time to leave because $t' > t_2'$. With a delay of t' the minimum distance is $d_{min,wait}$.

Since we perform this analysis when there is a loss of separation at the time of closest approach without delay and no current loss of separation, we can see graphically that t_a' is negative. Since a negative delay is not a valid solution in our case it is ignored. Moreover, it can be seen that if $d_{min,wait} < d_{min}$ there can be no solution as $t_2' < t'$

Obviously there is no solution if the current distance between the two agents is less than d_{min}. The solution is degenerate if $\mathbf{V_B} = -\mathbf{V_A}$. Indeed, in that case $\Delta\mathbf{V}$ is collinear with $-\mathbf{V_B}$, hence the minimum distance does not change with delays. The minimum distance is also fixed if $\mathbf{V_B}$ is null.

If agent A is on the ground or at a different altitude while it waits, the minimum separation while waiting does not matter. The solution is given by the latest intersection between the $\mathbf{P_A} - \mathbf{V_B}t$ line and the orange circle.

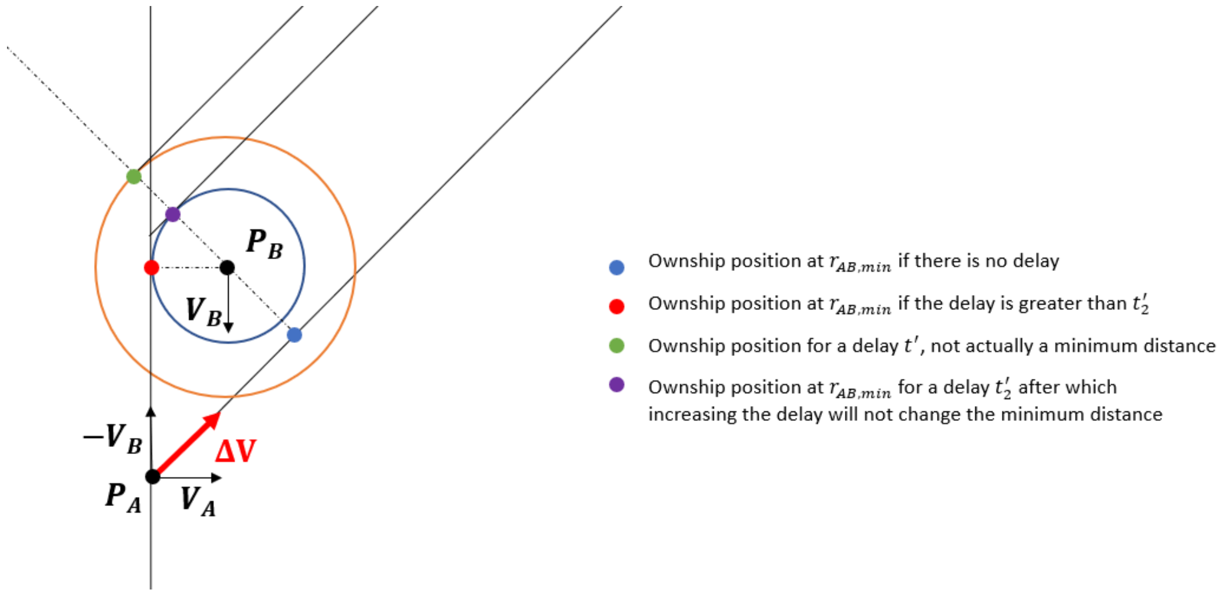
P_B
V_B
$-V_B$
ΔV
P_A
V_A
Ownship position at $r_{AB,min}$ if there is no delay
Ownship position at $r_{AB,min}$ if the delay is greater than t_2'
Ownship position for a delay t', not actually a minimum distance
Ownship position at $r_{AB,min}$ for a delay t_2' after which increasing the delay will not change the minimum distance

Fig. 14 Example of a situation where there is no valid time to leave

Acknowledgment

This research was supported in part through research cyberinfrastructure resources and services provided by the Partnership for an Advanced Computing Environment (PACE) at the Georgia Institute of Technology, Atlanta, Georgia, USA.

The authors would like to thank Tejas Puranik and Patrick Meyer for reviewing this paper.

References

[1] Gipson, L. N., "NASA Embraces Urban Air Mobility, Calls for Market Study," , 2018. URL `https://www.nasa.gov/aero/nasa-embraces-urban-air-mobility`, available at `https://www.nasa.gov/aero/nasa-embraces-urban-air-mobility`.

[2] Holden, J., and Goel, N., "Fast-Forwarding to a Future of On-Demand Urban Air Transportation," Tech. rep., Uber Elevate, San Francisco, 2016. URL `https://www.uber.com/elevate.pdf`.

[3] "Amazon Prime Air," , 2018. URL `https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011`, available at `https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011`.

[4] "Transforming the way goods are transported," , 2018. URL `https://x.company/projects/wing/`, available at `https://x.company/projects/wing/`.

[5] Crown Consulting Inc, Ascension Global, Georgia tech Aerospace Systems Design Lab, and McKinsey and Company, "UAM Market Study Executive Summary," Tech. rep., NASA, 2018. URL `https://www.nasa.gov/sites/default/files/atoms/files/uam-market-study-executive-summary-pr.pdf`.

[6] Booz Allen Hamilton, "Executive Briefing Urban Air Mobility Market Study," Tech. rep., 2018.

[7] NASA Aeronautics Research Mission Directorate, "Urban Air Mobility (UAM) Grand Challenge Industry Day," Tech. rep., NASA ARMD, 2018. URL `https://www.nasa.gov/uamgc`.

[8] Mavris, D. N., DeLaurentis, D. A., Bandte, O., and Hale, M. A., "A stochastic approach to multi-disciplinary aircraft analysis and design," *36th AIAA Aerospace Sciences Meeting and Exhibit*, 1998. https://doi.org/10.2514/6.1998-912.

[9] Sunil, E., and Hoekstra, J., "The Influence of Traffic Structure on Airspace Capacity," *Proceedings of the 7th International Conference on Research in Air Transportation*, 2016. https://doi.org/10.1108/13581981111106130, URL https://hal-enac.archives-ouvertes.fr/hal-01333624/.

[10] Zhu, G., and Wei, P., "Low-Altitude UAS Traffic Coordination with Dynamic Geofencing," *16th AIAA Aviation Technology, Integration, and Operations Conference*, , No. June, 2016. https://doi.org/10.2514/6.2016-3453, URL http://arc.aiaa.org/doi/10.2514/6.2016-3453.

[11] Peinecke, N., and Kuenz, A., "Deconflicting the urban drone airspace," *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, Vol. 2017-Septe, 2017. https://doi.org/10.1109/DASC.2017.8102048.

[12] Jang, D.-S., Ippolito, C. A., Sankararaman, S., and Stepanyan, V., "Concepts of Airspace Structures and System Analysis for UAS Traffic flows for Urban Areas," , 2017. https://doi.org/10.2514/6.2017-0449, URL http://arc.aiaa.org/doi/10.2514/6.2017-0449.

[13] Joulia, A. O., Dubot, T. O., and Bedouet, J. O., "Towards a 4D Traffic Management of Small UAS Operating at Very Low Level," *Proceedings of the 30th Congress of the International Council of the Aeronautical Sciences (ICAS2016)*, 2016, p. 2016_0064.

[14] Sachs, P., Dienes, C., Dienes, E., and Egorov, M., "Effectiveness of Preflight Deconfliction in High- Density UAS Operations," Tech. rep., Altiscope, 2018.

[15] Bulusu, V., Sengupta, R., Mueller, E. R., and Xue, M., "A Throughput Based Capacity Metric for Low-Altitude Airspace," *2018 Aviation Technology, Integration, and Operations Conference*, 2018, pp. 1–9. https://doi.org/10.2514/6.2018-3032, URL https://arc.aiaa.org/doi/10.2514/6.2018-3032.

[16] Sedov, L., and Polishchuk, V., "Centralized and Distributed UTM in Layered Airspace," *8th International Conference on Research in Air Transportation*, 2018, pp. 1–8.

[17] Johnson, M., Jung, J., Rios, J., Mercer, J., Homola, J., Prevot, T., Mulfinger, D., and Kopardekar, P., "Flight Test Evaluation ofan Unmanned Aircraft System Traffic Management (UTM) Concept for Multiple Beyond-Visual-Line-of-Sight Operations," *Proceedings of the Twelfth USA/Europe Air Traffic Management Research and Development Seminar (ATM2017)*, 2017.

[18] Keller, R. M., "Ontologies for aviation data management," *AIAA/IEEE Digital Avionics Systems Conference - Proceedings*, Vol. 2016-Decem, 2016. https://doi.org/10.1109/DASC.2016.7777971.

[19] Lee, H., and Bilimoria, K., "Properties of air traffic conflicts for free and structured routing," *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2001. https://doi.org/10.2514/6.2001-4051, URL http://arc.aiaa.org/doi/10.2514/6.2001-4051.

[20] Klooster, J., Torres, S., Earman, D., Castillo-Effen, M., Subbu, R., Kammer, L., Chan, D., and Tomlinson, T., "Trajectory synchronization and negotiation in Trajectory Based Operations," *29th Digital Avionics Systems Conference*, IEEE, 2010, pp. 1.A.3–1–1.A.3–11. https://doi.org/10.1109/dasc.2010.5655536.

[21] Federal Aviation Administration, "Introduction to TCAS II," Tech. rep., FAA, 2011. URL http://www.faa.gov/documentLibrary/media/Advisory{_}Circular/TCASIIV7.1Introbooklet.pdf.

[22] Hoekstra, J. M., and Ellerbroek, J., "BlueSky ATC Simulator Project: an Open Data and Open Source Approach," *7th International Conference on Research in Air Transportation*, 2016, pp. 1–8. https://doi.org/10.1109/TPC.2016.2516638, URL http://www.icrat.org/icrat/seminarContent/2016/papers/5/ICRAT{_}2016{_}paper{_}5.pdf.

[23] U.S. Department of Transportation, "Near Midair Collision Reporting," *Aeronautical Information Manual*, Aviation Supplies and Academics, Inc., 2018, Chaps. 7-6-3, p. 997.

[24] Vascik, P. D., and Hansman, R. J., "Scaling Constraints for Urban Air Mobility Operations: Air Traffic Control, Ground Infrastructure, and Noise," *2018 Aviation Technology, Integration, and Operations Conference*, 2018, pp. 1–25. https://doi.org/10.2514/6.2018-3849, URL https://arc.aiaa.org/doi/10.2514/6.2018-3849.

[25] Russel, S. J., and Norvig, P., "A* search: Minimizing the total estimated solution cost," *Artificial Intelligence: a Modern Approach*, Prentice Hall, 2009, Chap. 3, third edit ed., pp. 93–99.

[26] Phillips, M., and Likhachev, M., "SIPP: Safe Interval Path Planning for Dynamic Environments," *IEEE International Conference on Intelligent Robots and Systems*, IEEE, 2012, pp. 4708–4715. https://doi.org/10.1109/IROS.2012.6386191.

[27] Wanngren, N., "Dynamove – Multi Agent Navigation using Velocity Obstacles," Master, KTH, 2011.

[28] Van Den Berg, J., Guy, S. J., Lin, M., and Manocha, D., "Reciprocal n -Body Collision Avoidance," *Springer Tracts in Advanced Robotics*, 2011, pp. 3–19.

[29] Schouwenaars, T., De Moor, B., Feron, E., and How, J., "Mixed integer programming for multi-vehicle path planning," *Proceedings of European Control Conference*, , No. 3, 2001, pp. 2603–2608.

[30] Gurobi Optimization, L., "Gurobi Optimizer Reference Manual," , 2019. URL http://www.gurobi.com.

[31] Eby, M. S., "A Self-Organizational Approach for resolving Air Traffic Conflicts," *The Lincoln Laboratory Journal*, Vol. 7, No. 2, 1994, pp. 239–253.

[32] Maas, J., Sunil, E., Ellerbroek, J., and Hoekstra, J. M., "The Effect of Swarming on a Voltage Potential-Based Conflict Resolution Algorithm," *Proceedings of the 7th International Conference on Research in Air Transportation*, 2016. URL https://www.researchgate.net/profile/Emmanuel{_}Sunil/publication/304381434{_}The{_}Effect{_}of{_}Swarming{_}on{_}a{_}Voltage{_}Potential-Based{_}Conflict{_}Resolution{_}Algorithm/links/576d8f8a08ae6219474246a3.pdf.

[33] Snape, J., van den Berg, J., Guy, S. J., and Manocha, D., "The Hybrid Reciprocal Velocity Obstacle," *IEEE Transactions on Robotics*, Vol. 27599, No. October, 2009, pp. 1–11.

[34] Durand, N., and Barnier, N., "Does ATM Need Centralized Coordination? Autonomous Conflict Resolution Analysis in a Constrained Speed Environment," *Air Traffic Control Quarterly*, Vol. 23, No. 4, 2017, pp. 325–346. https://doi.org/10.2514/atcq.23.4.325.