ORCA-A* : A Hybrid Reciprocal Collision Avoidance and Route Planning Algorithm for UAS in Dense Urban Areas

Téo Chauvin, David Gianazza and Nicolas Durand
Fédération ENAC ISAE-SUPAERO ONERA
Université de Toulouse
Toulouse, France
teo.chauvin@enac.fr

***Abstract*—The rapid development of drones (or Unmanned Aerial Systems) and their potential deployment in urban areas poses a number of safety issues. Some degree of automation is most probably necessary to ensure that the UAS missions are safely and efficiently performed in urban environments. In a context where a large number of non-cooperative, non-communicative UAS would fly in dense urban areas, decentralized and autonomous approaches naturally come to mind. In such approaches, each agent would navigate among the buildings while avoiding the other traffic. ORCA (Optimal Reciprocal Collision Avoidance) is a state-of-the art geometric method for robot collision avoidance that could be used as a Detect & Avoid logic on-board UAS. It was initially designed for the 2D-motion of holonomic robots and requires some adaptation in order to be applied to flying objects in an urban environment. In particular, ORCA is a short-term collision avoidance that is not designed for path planning in a complex urban environment.**

In this study, we introduce a hybrid method combining ORCA with an A^* path-planning algorithm and show that ORCA-A^* significantly reduces the separation losses when compared with the baseline ORCA in artificial scenarios of dense UAS traffic.

***Keywords*—Detect and Avoid, U-Space separation management, Conflict resolution, Urban environment, Decentralized multiagent pathfinding,**

I. Introduction

The integration of drones into a wide array of tasks and contexts is one of the foremost challenges in modern robotics. Drones have the potential to revolutionize numerous applications, including surveillance, freight and passenger transport, civil security, and even military operations, thanks to their affordability, maneuverability, and the continuous advancements in their carrying capacity and autonomy. However, a critical requirement for drones is the ability to safely navigate from an initial point to a target destination. This is particularly challenging in a multi-agent environment where communication between drones is not possible. In such scenarios, ensuring safety hinges on the ability of each drone to independently detect and avoid collisions with other drones in the shared airspace.

State-of-the-art collision avoidance in robotics include geometric approaches based on Reciprocal Velocity Obstacles (RVO), such as ORCA (Optimal Reciprocal Collision Avoidance) [1]. For each pair of robots, a reciprocal velocity obstacle (RVO) materializes the area into which the relative velocity must not fall in order to avoid collision within a given time horizon τ. With both robots sharing the effort to avoid collision, each one implements a constraint on its velocity so that the relative velocity moves outside the RVO. In a multi-agent context, the RVOs of multiple pairs of robots define a convex polygon of constraints on the velocity of each agent.

ORCA is a reactive method where each agent takes its decisions based solely on a) the current positions and velocities of all the agents, b) its preferred direction (toward the destination). Assuming all agents share the same view of the traffic, ORCA provides coordinated lateral maneuvers that guarantee conflict-free t rajectories w ithin t he n ext τ s econds, without requiring communication among the agents.

Introducing obstacles (such as buildings, in an urban environment) can be done by adding specific velocity constraints. However, ORCA does not plan the robot's best possible route avoiding the obstacles. The robot might get stuck behind large obstacles obstructing its path toward the destination, or take a longer path than necessary.

In this paper, we propose to combine ORCA with an A^* algorithm. In this hybrid approach, the A^* algorithm operates on the visibility graph defined b y t he o bstacles, a nd finds the best route to the destination, and ORCA is in charge of avoiding collisions with the other agents while following the successive waypoints calculated by A^*. This preliminary study focuses on lateral trajectory deviations only, considering that all UAS fly a t a s ame, l ow a ltitude a mong t he buildings.

The rest of the paper is organized as follows. Related works are presented in Section II. The ORCA and A^* algorithms are briefly presented in sections III and IV respectively. Section V details the proposed method combining the reactive ORCA algorithm with the A^* shortest path algorithm. Section VI describes the parameterization of this new method and the design of scenarios used to compare ORCA-A^* with the baseline ORCA. The results of this comparison are presented in Section VII, before concluding in Section VIII.

II. Related works

A. Centralized vs. decentralized methods

A number of methods have been proposed in the literature to deconflict trajectories and avoid collisions between flying objects (aircraft, Unmanned Aerial Systems). Some rely on a centralized approach where deconflicting trajectories is addressed as a global optimization problem, using various methods such as MILP (*Mixed Integer Linear Program*) [2], semidefinite programming [3], genetic algorithms [4], or ant colony optimization algorithms [5]. These approaches provide high-quality solutions. However, they require an omniscient view of the environment in which the flying agents operate, as well as knowledge of their intentions (flight plans).

Other approaches are decentralized, with each flying agent deciding of its own maneuvers to avoid collisions with the other agents, based on what they perceive of the environment (obstacles, positions and velocities of the other agents).

Among these, reactive approaches select the action to be performed at an instant t based solely on the observation of the environment at that moment. They limit the planning horizon to the next step only. Two representatives of these approaches are APF (*Artificial Potential Fields*) and methods based on RVO (*Reciprocal Velocity Obstacles*).

APF are the most well-documented reactive approaches. They involve designing a potential function, often by analogy with physical phenomena, such as electrostatics [6] [7] [8]. Each mobile agent and obstacle induce an influence on all others. Obstacles exert a repulsive force, while the target and the other agents exerts an attractive force. These methods initially suffered from local minima in the potential functions, which could lead to deadlocks. New approaches have been developed to avoid these pitfalls. They are based on biharmonic functions, such as those inspired by fluid mechanics in [9], [10], and [11].

The use of RVO dates back to the *Relative Velocity Paradigm* by Fiorini and Shiller in 1993 [12]. The ORCA (*Optimal Reciprocal Collision Avoidance*) algorithm [1] is the state-of-the-art in geometric reactive algorithms based on RVO. Currently, there are many variations of ORCA: AVO (*Acceleration-Velocity Obstacle*) [13] computes solutions based on acceleration; NH-ORCA (*Non-holonomic ORCA*) [14] handles the case of non-holonomic drones; VR-ORCA (*Variable-Responsibility ORCA*) [15] ensures optimal sharing of avoidance responsibilities between two drones; CS-ORCA (*Constant Speed ORCA*) [16] addresses cases with low speed variation; DH-ORCA (*Dual-Horizon ORCA*) [17] proposes adding a decision time horizon to ORCA to limit the occurrence of pathological trajectories.

Finally, there are approaches based on deep learning, particularly reinforcement learning algorithms such as Deep-Q Learning [18].

Reactive methods are of particular interest in this study. In an operational context, they allow for rapid calculations based on the observation of the environment at time t. Reactive methods are well-suited for implementing autonomous and embedded Detect and Avoid algorithms. In particular, this paper focuses on the ORCA method. ORCA is reactive, geometric, and decentralized.

B. Motion-planning in discretized environments

Solving a motion planning problem involves determining a sequence of states (e.g. successive positions and velocities) that allows the mobile agent to reach its target in the most efficient way. A brute force algorithm is impractical given the large size of the solution space. The planning problem is, in the best case, PSPACE-complete [19]. It is necessary to use methods that accelerate the exploration of the solution space, typically by discretizing the space, and possibly using stochastic approaches in exploring the discretized space.

There are many different discretization methods: grid-based and graph-based. These involve selecting a set of nodes that represent the entire configuration space; cell-based methods, such as trapezoidal and rectangular decompositions [20] or CDT (*Constrained Delaunay Triangulation*) [21]; the most sophisticated methods are navigation meshes like LCT (*Local Clearance Triangulation*) [22] and ECD (*Explicit Corridor Map*) [23]. The underlying principle is always to simplify the representation of the space and thus reduce the size of the solution space.

Stochastic methods involve sampling the configuration space more efficiently. They converge in probability to the optimal solution and are widely used in practice for high-dimensional problems. Common examples include: RRT (*Rapidly Exploring Random Trees*) [24] and its variants like RRT* [25], RRT-Connect [26]; PRM (*Probabilistic Road Map*) [27]; RPF (*Random Potential Fields*) [28]; The state-of-the-art in stochastic methods is currently FMT (*Fast Marching Trees*) [29] due to their faster convergence speed.

C. Our hybrid approach

In this paper, we are interested in planning safe routes for UAS flying in an urban environment, while avoiding collisions with one another and with obstacles. We assume no communications and no explicit cooperation among the agents. We simply assume that every agent has a perfect and total view of the positions and velocities of the other agents, and that all agents apply the same anti-collision logic. Moreover, measure uncertainties are not taken into account and only lateral maneuvers are considered in this study.

Our approach to solve this decentralized, non-cooperative multi-agent motion planning problem consists in combining a path-finding A^* algorithm with a geometric reactive method, ORCA. The chosen space discretization on which the A^* operates to find a path for each UAS is the visibility graph defined by the obstacles. ORCA is in charge of avoiding collisions with the other UAS, and is slightly modified to take account of the route determined by the A^* path-planning algorithm. In our work, ORCA is the tactical short-term planner. It is the final decision layer that returns the velocity vector aiming at the next waypoint while avoiding other flying

agents. The A* algorithm combined with the visibility graph is the strategical long term decision layer that returns the overall path to reach the drone goal. Our hybrid method is therefore using both tactical and strategical layers to perform collision avoidance. This idea is already part of the VLL (Very Low Level) UAS geographical zones regulation. Z-Volumes, in particular, are designed to ensure high-density traffic and provide both in-flight tactical and pre-flight strategical collision avoidance services. The key difference is that our work focuses on a decentralized process that should be more flexible and scalable. The next three sections describe these methods and how we hybridize them.

III. ORCA

ORCA, which stands for Optimal Reciprocal Collision Avoidance [1], is a geometric avoidance algorithm for holonomic agents. That is, agents that can, at any given time t, choose a velocity vector in any direction. ORCA uses velocity obstacles to ensure collision avoidance between moving robots.

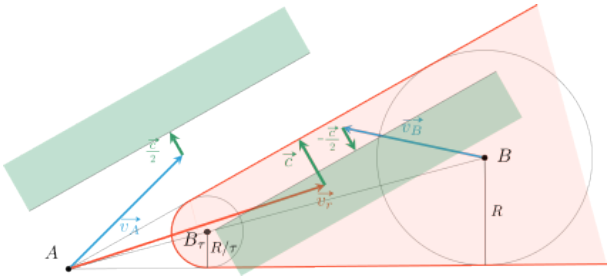


Figure 1. ORCA velocity obstacle (light red) and admissible velocity domains (in green) for two agents A and B

The situation is illustrated in Figure 1, considering a pair of agents A and B at a time t_0. Without loss of generality, we can assume $t_0 = 0$ in the following.

At time $t_0 = 0$, agent A and agent B have velocities $\vec{v_A}$ and $\vec{v_B}$, respectively. In Figure 1, the safety distance to be maintained is R, so the agent A must never be inside the circle $D(B, R)$ centered at B with radius R. Let us consider the moving reference frame attached to the agent B, and the relative velocity of A with respect to B, denoted $\vec{v_r} = \vec{v_A} - \vec{v_B}$.

The relative velocity vector leads to a loss of separation between A and B within a time horizon τ if and only if for some $t \leq \tau$, $\vec{v_r} \cdot t \in D(B, R)$. Thus, if the relative velocity vector is within the disk $D(B_\tau, \frac{R}{\tau})$ and assuming that the relative velocity remains unchanged, then it can be asserted that there will be a loss of separation at τ seconds. This reasoning can be extended to any time $t < \tau$, resulting in the rounded cone shaded in light red in Figure 1. It represents the set of relative velocities $\vec{v_r}$ that lead to a loss of separation at some time within the time interval $[0, \tau]$.

In order to avoid collision, the velocities of both agents A and B must be changed so that the relative velocity $\vec{v_r}$ falls outside the forbidden zone (the rounded cone in light red). The smallest effort consists in moving $\vec{v_r}$ toward the closest boundary of the velocity obstacle. This move is represented by the vector $\vec{c}$ on Figure 1. The effort to move $\vec{v_r}$ outside the velocity obstacle is shared between the two agents A and B. The collision avoidance domain for the velocity $\vec{v_A}$ is the semi-plane (in green) defined by the parallel to the cone boundary and an offset of $\frac{\vec{c}}{2}$ from the tip of the vector $\vec{v_A}$. Assuming the agent B follows the same logic, a similar semi-plane is defined for the velocity $\vec{v_B}$, also shown in green on Figure 1. If both agents select their velocities in their respective collision avoidance domains, then ORCA guarantees that no collision will occur between A and B within the next τ seconds.

When more than two agents are involved, the collision avoidance domain of an agent A is simply the intersection of all the semi-planes determined by all the pairwise velocity obstacles in which A is involved. The intersection of these half-planes is a convex set.

In summary, at each time step, the agent A calculates its collision avoidance domain as explained above. Agent A then selects the velocity that is within this convex set and closest to its optimal velocity vector toward its destination. If all agents follow the same logic and share the same view of the environment, ORCA provides coordinated maneuvers to avoid collisions, without requiring communication among the agents.

IV. The A^* Algorithm

A* is a BFS (*Best First Search*) algorithm applied to a graph and guided by a heuristic which gives higher priorities to the exploration of the most promising nodes. Starting from an initial node in the graph and considering transition costs associated to the edges of the graph, the cost associated to a node is simply the sum of the transition costs that led to this node, following the best path found so far.

These costs are updated during the search as follows.

The algorithm handles three sets of nodes: the expanded nodes E, the terminal nodes T, and the frontier nodes F. Let u_0 denote the initial node. The goal is to reach one of the nodes in T. The frontier corresponds to a set of nodes for which the priority has been calculated, but which have not yet been visited. At each step, A* considers the node in F with the highest priority, which is the most promising direction to search. The priority of a node v in F, denoted $f(v)$, is defined as the sum of the cost associated with the node v and a heuristic $h(v)$.

The cost associated with node v, denoted $cost_v$, is the best cost found so far to reach v from u_0. Let $k(u, v)$ be the transition cost from node u to node v. If u is the parent node of v (*i.e.* u is the preceding node in the best path found so far to v), then $cost_v = cost_u + k(u, v)$. The parent of each node is also memorized, together with the costs, and updated when necessary. The heuristic, denoted $h(v)$, provides an "estimate" of the cost to move from node v to the nearest terminal node.

Typically, for path-planning in 2D geographic environments, the nodes represent geographic positions, the transition cost $k(u, v)$ is the distance between positions u and v, the cost

$cost_v$ is the length of the best path found so far from u_0 to v, and the heuristic $h(v)$ is the distance as the crow flies from v to the terminal node (the destination).

Finally, the priority is given by $f(v) = -cost_v - h(v)$. This corresponds to the estimated cost to go from the starting point to the destination point via node v. The negative sign is used because higher priority is given to lower costs; extracting the node with the highest priority is equivalent to extracting the node with the minimal cost.

The A* algorithm extracts the node u with the highest priority from the frontier F, develops this node by considering its neighbors. For each neighbor v, the cost of the path to v passing by u is computed and compared with the previous cost of v, if any. If the cost is improved, the new value is stored and u is memorized as the parent of v. If the neighbor has never been visited before, its priority is computed, and it is inserted in the frontier. The algorithm stops when a terminal node is visited. The cost of this terminal node is minimal, and the best path is obtained, retrieving the parent node and its predecessors. Optimality is guaranteed by the fact that the heuristic is consistent (always shorter than the real distance).

V. The hybrid ORCA-A^* algorithm

Reactive methods are effective, notably because they do not anticipate future actions. Anticipation allows for achieving better global solutions. The optimal decision at time t depends on decisions at subsequent times, and thus on the overall final trajectory. Therefore, adding a planning layer can help guide the reactive algorithm towards better quality solutions.

In particular, ORCA allows reaching a point while ensuring avoidance of both static and dynamic obstacles. We propose to relieve ORCA of the responsibility of avoiding static obstacles, so that it only handles dynamic obstacles. In other words, obstacles do not raise additional constraints inside the ORCA logic. Planning then consists of providing a set of intermediate points to reach the goal. Each of these intermediate points is placed such that the drone can follow a direct trajectory from one to another without worrying about the presence of static obstacles. In other words, the segments connecting successive intermediate points are in the free space, away from obstacles. The long term planning is computed by each drone only once, at $t = 0$. During the simulation, ORCA might perform an avoiding maneuver to ensure safety. This locally modified trajectory differs from the originally planned trajectory, and thus may not be compatible with the set of pre-computed waypoints. If the targeted waypoint is out of sight, then the entire long-term planning is actualized, starting from the current position.

The reactive ORCA algorithm is responsible for the control, i.e., detecting and avoiding dynamic obstacles. Meanwhile, planning is responsible for navigation; it only concerns the route to follow and considers only static obstacles. This can also be referred to as short-term and long-term planning. The long-term planning algorithm we use is the A* shortest path algorithm. We seek the shortest route in the space.

The A* algorithm operates on a representation of space in the form of a graph. Thus, at each node, a finite number of successors can be defined, and at each edge, a cost can be defined. In our case, we use the visibility graph as shown in figure 2. Each node corresponds to one of the vertices of the polygonal obstacles, and the costs associated with the edges are the Euclidean distances between nodes. In a visibility graph, an edge exists between two nodes u_i and u_j if and only if the segment (u_i, u_j) is in the free space. This property of the visibility graph justifies its use in our context.

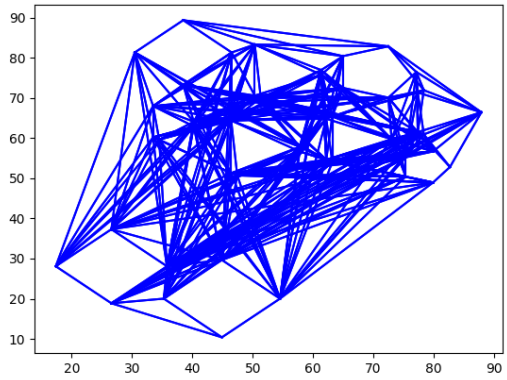


Figure 2. Example of a visibility graph

VI. Experiment setup

ORCA depends on several parameters. The time horizon τ corresponds to the duration during which an agent assumes that the behavior of other observable agents remains constant. A long time horizon means considering separation losses that would occur far into the future, provided that the behavior of the involved drones does not change. This leads to small-amplitude maneuvers that may prove unnecessary or counterproductive. A short time horizon implies that the behavior of other agents is unpredictable, leading to necessary abrupt maneuvers. The ideal value for the time horizon might depend on the number of agents, their dynamic and kinematic characteristics, their positions and velocities, and the very nature of the environment. It must be adjusted on scenarios that are representative of the environment in which ORCA will operate.

The simulation parameters were chosen in collaboration with the ENAC UAS team, as we plan to validate our hybrid algorithm on real UAS in a small scale environment. The environment is a square with a side length of 100m. The imposed separation distance between agents is 3m. The maximum speed of the agents is 8.33m/s, or 30km/h. Inertial effects are not considered. Each simulation starts from a scenario that specifies the position and shape of obstacles, the starting and ending points of the agents, and the time at which they enter the zone. Three types of scenarios are distinguished: empty scenarios with no static obstacles; sparse scenarios with a few

small-sized obstacles spaced apart; and urban scenarios with large obstacles separated by roads. Each of these scenarios can be simulated with varying numbers of agents. We considered three levels of densities: 1) low density with one agent entering the zone every two seconds; 2) medium density with one agent every second; 3) high density with two agents every second. In total, there are nine possible configurations.

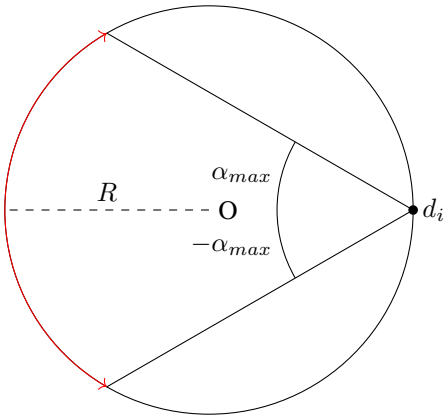


Figure 3. Choice of departure and destination points.

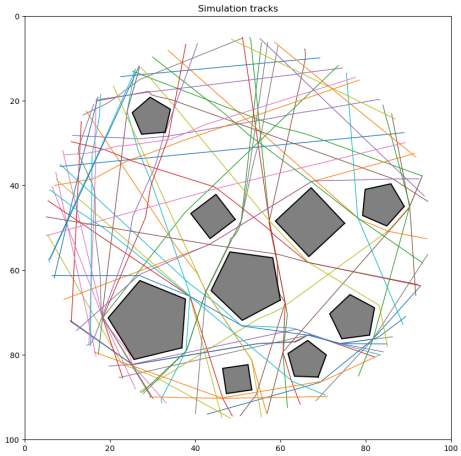
To ensure that each agent crosses the obstacle zone, agents appear on a circle centered at O, the center of the environment, with a radius of $R = 45m$. Obstacles are placed inside the circle. For each agent, a starting point d_i on the circle is chosen uniformly. The starting direction is drawn uniformly, which is the vector oriented towards the center, deviated by an angle drawn uniformly between $-\alpha_{max}$ and α_{max}. The arrival point is the intersection of the ray directed by the vector and the circle of radius R (see Figure 3). The mobiles appear on the circle at random times. The probability of a mobile appearing follows a Poisson distribution with parameter λ. The expectation is λ, which is the number of mobiles appearing per unit of time, known as the incoming flow.

We simulated, for each configuration, five values of τ ranging from one to five seconds looking for a parameter independent of the configuration. We found that for the ORCA algorithm without planning, the best parameterization was $\tau = 3s$. For ORCA with planning using A* on the visibility graph, the best parameterization was $\tau = 1s$.

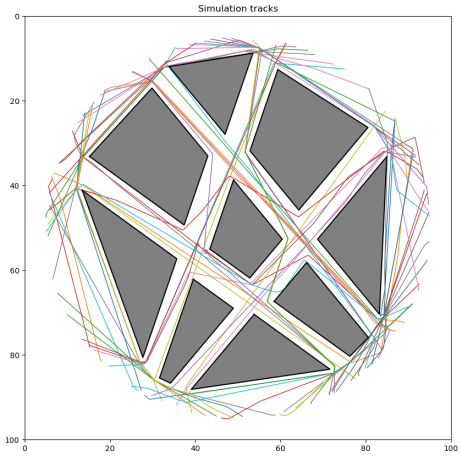
Figures 4a and 4b present examples of ORCA-A* simulations made on both sparse and urban environments, respectively, for an intermediate incoming flow.

VII. Results

With the selected parameters for ORCA and ORCA-A^*, we compared the two methods across nine different random configurations, considering empty, sparse, and urban environments with low, medium and high traffic densities. The results of the empty, sparse and urban environments are shown in Figure 5. Each figure shows the total time spent in loss of separation (LoS) for low, medium, and high densities.



(a) ORCA-A* in ”sparse” environment



(b) ORCA-A* in ”urban” environment

Figure 4. Examples of environments

When there are no static obstacles, ORCA results in fewer losses of separation compared to ORCA-A*. Indeed, ORCA and ORCA-A* are then essentially the same algorithms, differing only in the parameterization of the time horizon. The losses of separation are lower with ORCA because it anticipates and avoids potential collisions in advance.

When static obstacles are introduced, whether in sparse or urban environments, the performance of ORCA-A* is significantly better, the more so with dense traffic in complex environments. Although ORCA can reach its goal point in most cases by considering constraints imposed by static obstacles, this approach is less effective than adding a long-term planning layer.

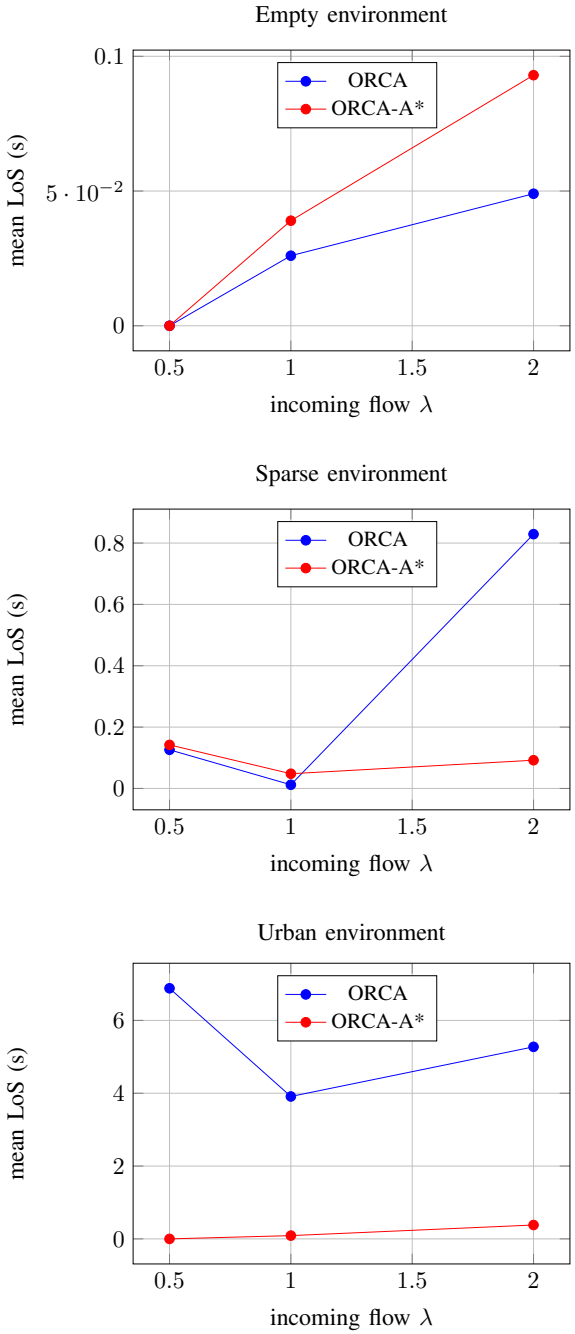


Figure 5. Simulation results

VIII. Conclusion

To summarize, we proposed to combine a short-term collision avoidance method, ORCA, with a long-term path-planning A^* algorithm. This hybrid approach showed better results in terms of separation losses than the standard ORCA algorithm, on the nine scenarios on which it was tested. The improvement was most significant on dense urban scenarios, showing the interest of long-term planning in this context. Our work shows that, at least under simple hypotheses, adding a pre-flight strategical decision layer on top of ORCA logic increases the original ORCA performance, and even on a completely decentralized process without any information shared between flying drones.

In future works, we plan to test our approach on a larger number of random scenarios, and to confirm our first results with some statistical evidence. We also plan to compare our method with other reactive methods. Another interesting question that we intend to investigate is: What would be the potential benefits of sharing information among the agents, and what information should be shared? Clearly, planning a route without knowing the intentions of the other agents is sub-optimal, and could be improved knowing the destinations of the other agents, for example. In this study, we have considered 2D lateral maneuvers only, assuming all UAS fly at the same altitude. One could imagine organizing the UAS traffic in urban areas in several horizontal layers. The ORCA algorithm and its variants would then require some adaptation to handle altitude changes in addition to lateral maneuvers.

Finally, other key issues must be addressed prior to any operational deployment of collision avoidance methods such as ORCA or any other Detect & Avoid (D&A) methods. Notably, the robustness of D&A to an imperfect or partial perception of the environment is crucial to ensure safe operations in urban environments.

References

[1] M. L. Jur van den Berg Stephen J. Guy and D. Manocha, "Optimal reciprocal collision avoidance for multi-agent navigation," *Proceedings IEEE International Conference on Robotics and Automation*, 2010.

[2] L. Pallottino and A. Bicchi, "Mixed integer programming for aircraft conflict resolution," *American Institute of Aeronautics and Astronautics*, 2001.

[3] J.-H. O. E. Frazzoli Z.-H. Mao and E. Feron, "Resolution of conflicts involving many aircaft via semidefinite programming," *Journal of guidance, control, and dynamics*, 2001.

[4] N. D. Géraud Granger and J.-M. Alliot, "Token allocation strategy for free-flight conflict solving," *13th Conference on Innovative Applications of Artificial Intelligence*, 2001.

[5] N. Durand and J.-M. Alliot, "Ant colony optimization for air traffic conflict resolution," *ATM Seminar 2009, 8th USA/Europe Air Traffic Management Research and Development Seminar, Jun 2009, Napa, California, United States*, 2009.

[6] O. Khatib, "Real-time obstacle avoidance for manipulators ad mobile robots," *The International Journal of Robotics Research*, 1986.

[7] M. S. Eby and W. E. Kelly, "Free flight separation assurance using distributed algorithms," 1999.

[8] J. K. C. T. G. Pappas and S. Sastry, "2-1/2 d conflict resolution maneuvers for atms," *Proceedings of the 37th IEEEConference on Decision ControlTampa, Florida USA*, 1998.

[9] D. Keymeulen and J. Decuyper, "The fluid dynamics applied to mobile robot motion:the stream field method," 1994.

[10] C. Louste and A. Liégeois, "Path planning for non-holonomic vehicles: A potential viscous fluid field method," *Robotica volume 20 p.291-298*, 2002.

[11] I. Y. Zeynep Bilgin Murat Bronz, "Panel method based guidance for fixed wing micro aerial vehicles," *13th I NTERNATIONAL M ICRO A IR V EHICLE C ONFERENCE*, 2022.

[12] Z. S. Paolo Fiorini, "Motion planning in dynamic environments using the relative velocity paradigm," *Proceedings IEEE International Conference on Robotics and Automation, Atlanta, GA, USA, 1993, pp. 560-565 vol.1, doi: 10.1109/ROBOT.1993.292038*, 1993.

[13] S. J. G. Jur van den Berg Jamie Snape and D. Manocha, "Reciprocal collision avoidance with acceleration-velocity obstacles," *IEEE International Conference on Robotics and Automation, Shanghai, China, 2011, pp. 3475-3482, doi: 10.1109/ICRA.2011.5980408.*, 2011.

[14] M. R. Javier Alonso-Mora Andreas Breitenmoser, P. Beardsley, and R. Siegwart, "Optimal reciprocal collision avoidance for multiple non-holonomic robots," *Martinoli, A., et al. Distributed Autonomous Robotic Systems. Springer Tracts in Advanced Robotics, vol 83. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-32723-0*$_1$5, 2013.

[15] T. F. Ke Guo Dawei Wang and J. Pan, "Vr-orca : Variable responsability optimal reciprocal collision avoidance," *IEEE Robotics and Automation Letters, vol. 6, no. 3, pp. 4520-4527, July 2021, doi: 10.1109/LRA.2021.3067851*, 2021.

[16] N. Durand, "Constant speed optimal reciprocal collision avoidance," *Transportation Research Part C: Emerging Technologies (2018) volume 96, 366-379*, 2018.

[17] D. Alligier Gianazza and al., "Dual-horizon reciprocal collision avoidance for aircraft and unmanned aerial systems," *J Intell Robot Syst 107, 3 (2023). https://doi.org/10.1007/s10846-022-01782-2*, 2023.

[18] H. Bae, G. Kim, J. Kim, D. Qian, and S. Lee, "Multi-robot path planning method using reinforcement learning," *Applied Sciences 2019, 9(15), 3057; https://doi.org/10.3390/app9153057*, 2019.

[19] T. Bylander, "The computational complexity of propositional strips planning," *Artificial Intelligence,Volume 69, Issues 1–2,1994,Pages 165-204,ISSN 0004-3702,https://doi.org/10.1016/0004-3702(94)90081-7.*, 1994.

[20] C. M. Ramon Gonzalez Marius Kloetzer, "Comparative study of trajectories resulted from cell decomposition path planning approaches," *21st International Conference on System Theory, Control and Computing (ICSTCC)*, 2017.

[21] Y. C. Hongyang Yan Huifang Wang and G. Dai, "Path planning based on constrained delaunay triangulation," *Proceedings of the 7th World Congress on Intelligent Control and Automation, June 25 - 27, 2008, Chongqing, China*, 2008.

[22] M. Kallmann, "Shortest paths with arbitrary clearance from navigation meshes," *Proceedings of the Eurographics/SIGGRAPH Symposium on Computer Animation, Madrid, Spain*, 2010.

[23] R. Geraerts, "Planning short paths with clearance using explicit corridors," *IEEE International Conference on Robotics and Automation Anchorage Convention District, May 3-8, 2010, Anchorage, Alaska, USA*, 2010.

[24] S. M. LaValle, "Rapidly exploring random trees : A new tool for path planning," *The annual research report*, 1998.

[25] E. F. Sertac Karaman, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, 2011.

[26] J. J. K. Jr. and S. M. Laval, "Rrt-connect : An efficient approach to single-query path planning," *International Conference on Robotics Automation*, 2000.

[27] M. Kallmann and M. Mataric', "Motion planning using dynamic roadmaps," *International Conference on Robotics 8 Automation*, 2004.

[28] S. Huang, "Path planning based on mixed algorithm of rrt and artificial potential field method," *The 4th international conferenceon intelligent robotics and control engineering*, 2021.

[29] J. L, S. E, C. A, and P. M., "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *The International Journal of Robotics Research. 2015;34(7):883-921. doi:10.1177/0278364915577958*, 2015.