drones

MDPI

Article

Path Planning for Fixed-Wing Unmanned Aerial Vehicles: An Integrated Approach with Theta* and Clothoids

Salvatore Rosario Bassolillo [1], Gennaro Raspaolo [2], Luciano Blasi [2], Egidio D'Amato [3,*] and Immacolata Notaro [2]

[1] Institute for High Performance Computing and Networking of National Research Council (ICAR-CNR), Via Pietro Castellino 111, 80131 Naples, Italy; salvatore.bassolillo@icar.cnr.it
[2] Department of Engineering, University of Campania Luigi Vanvitelli, 81031 Aversa, Italy; gennaro.raspaolo@unicampania.it (G.R.); luciano.blasi@unicampania.it (L.B.); immacolata.notaro@unicampania.it (I.N.)
[3] Department of Science and Technology, University of Naples Parthenope, 80143 Napoli, Italy
* Correspondence: egidio.damato@uniparthenope.it

Abstract: Unmanned Aerial Vehicles (UAVs) have emerged as a compelling alternative to manned operations, offering the capability to navigate hazardous environments without risks for human operators. Despite their potential, optimizing UAV missions in complex and unstructured environments remains a pivotal challenge. Path planning becomes a crucial aspect to increase mission efficiency, although it is inherently complex due to various factors such as obstacles, no-fly zones, non-cooperative aircraft, and flight mechanics limitations. This paper presents a path-planning technique for fixed-wing unmanned aerial vehicles (UAVs) based on the Theta* algorithm. The approach introduces innovative features, such as the use of Euler spiral, or clothoids, to serve as connection arcs between nodes, mitigating trajectory discontinuities. The design of clothoids can be linked to the aircraft performance model, establishing a connection between curvature constraints and the specific characteristics of the vehicle. Furthermore, to lower the computational burden, the implementation of an adaptive exploration distance and a vision cone was considered, reducing the number of explored solutions. This methodology ensures a seamless and optimized flight path for fixed-wing UAVs operating in static environments, showcasing a noteworthy improvement in trajectory smoothness. The proposed methodology has been numerically evaluated in several complex test cases as well as in a real urban scenario to prove its effectiveness.

Keywords: path planning; unmanned aerial vehicle; clothoids; obstacle avoidance

Citation: Bassolillo, S.R.; Raspaolo, G.; Blasi, L.; D'Amato, E.; Notaro, I. Path Planning for Fixed-Wing Unmanned Aerial Vehicles: An Integrated Approach with Theta* and Clothoids. *Drones* **2024**, *8*, 62. https://doi.org/10.3390/drones8020062

Academic Editors: Hanno Hildmann and Fabrice Saffre

Received: 30 December 2023
Revised: 2 February 2024
Accepted: 7 February 2024
Published: 12 February 2024

1. Introduction

Unmanned Aerial Vehicles (UAVs) have attracted significant attention as a viable substitute for manned operations. The ability to operate in hazardous environments without risks for human operators, combined with the cost-effectiveness of operations, has propelled UAVs into one of the most rapidly expanding domains. In fact, the number of drones and unmanned operations is expected to double by 2025 [1,2].

To fully exploit the potential of UAVs and improve mission efficiency, a key challenge is planning paths in complex and unstructured environments.

Nevertheless, path planning presents a challenge due to the inherent complexity of environments. This issue is commonly framed as an optimization problem, where the definition of the shortest path involves a sequence of waypoints. These waypoints must be chosen considering the presence of obstacles, no-fly zones, non-cooperative aircraft, and limitations from flight mechanics [3,4]

In the literature, several publications explored the definition of three-dimensional trajectories by decoupling planar maneuvers from altitude changes [5,6]. Extensive research

addressed two-dimensional (or decoupled three-dimensional) aerial path planning issues, sharing algorithms and solutions with the fields of robotics and automotive science.

Literature reviews on UAV path planning classify methods into variational, optimal control, geometrical, graph optimization, artificial potential field, and natural optimization. While the variational approach to path planning is considered natural [7,8], its application becomes challenging in complex scenarios with flight dynamics constraints.

Optimal control serves as an alternative, involving a closed-loop algorithm with convergence to a local sub-optimal trajectory. In [9], the authors addressed the challenge of flying a helicopter with a slung load by formulating an optimal control problem, using trajectories as references to a state feedback controller. Pseudospectral optimal control [10] was also used for unmanned helicopters, to plan minimum-time trajectory in environments with obstacles. The authors of [11] introduced a Distributed Model Predictive Controller (DMPC) for UAV guidance, taking into account anticollision constraints in accordance with the International Civil Aviation Organization (ICAO) Right of Way rules. However, optimal control may be ineffective in finding a global optimum, as it demands significant computational capabilities, especially when employing non-linear optimization algorithms.

Several publications [12,13] introduced purely geometrical approaches, describing paths as sequences of segments, arcs, or template curves. Examples include solutions to Dubins' car problem [14,15] or clothoid curves [16,17]. However, such solutions often require combining other algorithms to account for obstacles and no-fly zones.

Graph optimization is an effective strategy for designing edges using geometrical methods. In these approaches, the environment needs to be discretized using regular or irregular grids [18]. Alternatives include Visibility Graphs [19–21], Voronoi diagrams [22,23], Rapidly Exploring Random Trees (RRT) [24–26], Tangent Graphs [27], Sparse Tangential Networks (SPARTAN) [28], and road maps [29,30].

In order to determine the most efficient route across graphs, ensuring consistent and reproducible outcomes in expansive settings, widely recognized heuristics like Dijkstra's algorithm [31], A* [32,33], and D* [34] are frequently employed. However, these algorithms do not guarantee the shortest path in continuous space. Indeed, in [35], the Theta* approach was introduced, which outperformed several solutions based on A*, A* with post-smoothing [36], and the so-called Field D* [37], in terms of path length and computation time. The Theta* algorithm represents an improvement over A* and it is particularly useful in scenarios where agents need to find the shortest path through a continuous space. Theta* allows for a more flexible search by moving along directions, minimizing heading changes. Ref. [38] presented a grid map-based path-planning algorithm, based on Theta*, for the real-time generation of realistic paths, taking into account both the heading angle and the angular rate of unmanned surface vehicles (USVs). To improve A*, in [33], the authors proposed a Theta*-based procedure applied to orographic obstacles and urban environments, in order to evaluate the solution for different kinds of obstacles. In [39], a revised and adapted Lazy Theta* path-planning algorithm was presented, reducing the dimension of the problem and using a sparse resolution grid in the form of an octree, in order to generate paths in large 3D scenarios in real-time.

In robot motion planning problems, kinodynamic path searching methods have been adopted to satisfy kinematic and dynamic constraints. In particular, several hybrid A* algorithms [40–42] combine the benefits of A* search in continuous space with a discretized set of headings in order to obtain feasible trajectories for autonomous vehicles navigating in unknown environments. In [43], the authors proposed a hierarchical path-planning architecture for UAVs, consisting of an A* path-search step followed by B-Spline trajectory optimization. In [44], the authors proposed an improved A* and a path pruning method to generate safe guiding trajectories for quadrotor UAVs. Another example is the Kinodynamic RRT* (KRRT*) algorithm, first introduced in [45], where the authors extended the capabilities of RRT* to obtain smoother trajectories for robot motion planning with linear differential constraints. In [46], the authors presented an application of this approach, named Multi-robot Kinodynamic RRT* (MK-RRT*), to the path-planning problem for a

fleet of UAVs, testing their algorithm in real-world experiments. Alternatively, the authors in [47] employed the KRRT* to plan trajectories for fixed-wing UAVs in complex terrain environments.

In the context of guidance algorithms with real-time requirements, the techniques based on Artificial Potential Field have proven to be highly effective [48–50]. However, increasing the number of obstacles or potential sources in the environment raises the likelihood of encountering singularities, which need a strategy to find a solution.

Natural optimization methods enable the creation of sophisticated models [51,52] based on flight dynamics, where a sequence of feasible maneuvers is optimized to reach the target point. In [53], the authors proposed a technique for creating continuously parameterized classes of feasible trajectories, made of a small collection of user-provided example motions. An incremental path-planning algorithm for autonomous vehicles flying in an environment with moving obstacles able to take into account flight dynamics was presented in [54]. Furthermore, a trajectory planning model, coupled with a Particle Swarm Optimizer (PSO) [55] was able to identify optimal UAV flight trajectories compliant with environmental constraints. Nevertheless, these methods are computationally expensive and slow, making them only suitable for offline optimization applications.

This article introduces a novel approach to the path planning of a fixed-wing aircraft. The trajectory is defined as a piece-wise curve made of segments, circular and spiral arcs between two prescribed points, with assigned initial and target directions. To ensure compliance with aircraft performance constraints, the heading variation between two directions is made by defining an Euler spiral-based path transition curve made of two spirals and a circular arc, where the maximum curvature and the maximum change in curvature depend on the aircraft characteristics. The final path is optimized by using an evolution of the Theta* algorithm, whose arcs between nodes are the above defined path-transition curves. In this manner, the final trajectory is free of discontinuities and proves to be entirely flyable by the aircraft. This approach is an important enhancement with respect to previous works that adopt Dubins arcs to smooth the path, introducing a discontinuity in the curvature between straight segments and circular arcs. The transition between lines with discontinuous curvature requires a step in the yaw rate, and, consequently, in the bank angle, which is impossible due to flight dynamics constraints. To enhance the performance of the presented algorithm, the exploration distance of Theta*, which is useful to exploit the search space, is variable, taking into account the actual distance to obstacles. Additionally, in order to avoid unnecessary areas and prevent excessive maneuvers, Theta* explores the search space using a vision cone, the extent of which is a parameter of the algorithm.

The manuscript is structured as follows: Section 2 introduces the path transition curve as a solution to path planning in the absence of obstacles. Section 3 outlines the Theta*-based path-planning algorithm, incorporating the concept of smooth transitions between edges linking nodes. In Section 4, several numerical results are presented to prove the effectiveness of the proposed approach, incorporating a comparative analysis of several implemented versions of Theta*. Section 5 introduces a numerical comparison of the Theta*-based algorithm with three state-of-the-art planners in order to highlight the pros and cons of the proposed methodology. Finally, Section 6 provides a concise summary of the conclusions drawn from the current study.

2. Path Transition Curve

A standard flight path, navigating through a sequence of waypoints, comprises linear and circular trajectory segments [56]. In fixed-wing aircraft, the transition between these segments poses challenges due to inherent path curvature discontinuity, requiring an immediate shift in yaw rate and bank angle. Flight dynamics prerequisites stress the need for continuous curvature paths within specified bounds for precise tracking [57,58]. Euler spirals or clothoids, based on a linear 2D dynamic aircraft model, offer a linear curvature and arc length relationship, making them apt for geometrically determining the flight path during optimization.

The model simulates non-steady constant altitude turns for altering aircraft heading and estimating fuel consumption. This study focuses on coordinated turns with zero sideslip angle, where the velocity vector consistently lies in the plane of symmetry. Heading angle (ψ) and roll angular velocity (μ) are key parameters, and motion is confined to a horizontal plane.

Considering quasi-steady turns, the following simplifications can be employed: (a) a negligible change in velocity ($\dot{V} = 0$), (b) a negligible component of thrust normal to the flight path, and (c) a constant weight.

The angle of attack (α), drag (D), thrust (T), and specific fuel consumption (C) can be considered as functional relationships of the form:

$$\alpha = \alpha(h, V, L),\ D = D(h, V, L),\ T = T(h, V, P),\ C = C(h, V, P)$$

where h is the altitude and P is the engine power.

The equations of motion can be formulated in terms of speed (V) and heading angle (ψ) as follows:

$$\begin{cases} \dot{x} = V\cos\psi \\ \dot{y} = V\sin\psi \\ \dot{V} = 0 = (g/W)\cdot(T(h,V,P) - D(h,V,L)) \\ \dot{\psi} = gL\sin\phi/WV \\ \dot{\phi} = \mu \\ \dot{\gamma} = 0 = (g/WV)\cdot(L\cos\phi - W) \\ \dot{W} = -C(h,V,P)T(h,V,P) \end{cases} \tag{1}$$

These eight equations are based on ten variables (six states: $x, y, \psi, \phi\gamma, W$, and four controls: V, P, L, μ).

Considering subsonic speed, the drag polar can be approximated with a parabolic function with constant coefficients as follows:

$$C_D = C_{D_0} + KC_L^2 \tag{2}$$

where C_{D_0} and K are constants and C_L is the lift coefficient that can be computed as

$$C_L = \frac{L}{\frac{1}{2}\rho V^2 S} = \frac{W}{\frac{1}{2}\rho V^2 S\cos\phi} \tag{3}$$

From this model, it is possible to establish the curvature and the sharpness of the trajectory during a turn maneuver. Considering a typical circular motion of radius r, whose model is given by

$$r = \frac{W}{g}\frac{V^2}{F} \tag{4}$$

with the centripetal force $F = L\sin\phi$, the turn radius can be deduced as follows:

$$r = \frac{W}{g}\frac{V^2}{L\sin\phi} = \frac{2}{\rho g}\frac{W}{S}\frac{1}{C_L}\frac{1}{\sin\phi} \tag{5}$$

where ρ is the air density and $\frac{W}{S}$ represents the wing loading. However, because $L\cos\phi = W = \frac{1}{2}\rho V^2 S C_L\cos\phi$, the turn radius can be written as dependent on only the speed ($V = const$) and the bank angle ϕ.

At constant speed, the minimum turn radius r_{min} can be obtained when $\tan\phi$ is maximum, taking into account the maximum load factor n_{max} compliant with the maximum available thrust and the structural limits of the aircraft:

$$\tan\phi_{max} = \sqrt{n_{max}^2 - 1} \tag{6}$$

Because the curvature k is inversely proportional to the curvature radius, a path is valid if the curvature is smaller than the specified maximum value

$$|\kappa(t)| < \kappa_{max} = \frac{1}{r_{min}} = \frac{g \tan \phi_{max}}{V^2} \tag{7}$$

where k_{max} denotes the maximum curvature.

The maximum sharpness can be found by differentiating the curvature function with respect to time:

$$\sigma_{max} = \frac{g}{V^2} \dot{\phi}_{max} \sec^2 \phi_{max} \tag{8}$$

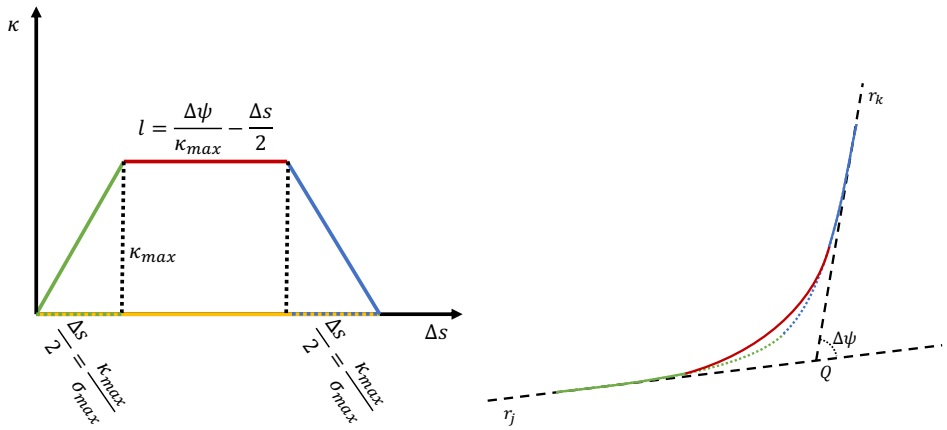
Since the curvature of a spiral is linear along the curve, the minimum and maximum curvature occur at the tips of the clothoid segment.

Flight Path Transition Curve between Two Directions

Consider two straight lines, denoted as r_j and r_k intersecting at a point Q, that define the desired headings for an aircraft, represented by ψ_j and ψ_k, respectively.

A flight path transition curve between these intersecting straight lines can be computed using two clothoids and an arc (coordinated turn), if necessary. The presence of such a circular arc must be taken into account, considering any constraints on the aircraft roll rate. This definition can be regarded as an alternative to the Dubins path [14], achieved by smoothing the tips of the circular arc to avoid curvature discontinuities.

With reference to Figure 1, achieving the transition from the linear trajectory, defined by r_j, to the circular arc involves a curvature increment from zero to k_{max}. Subsequently, the curvature remains constant until another transition occurs between the arc and the final direction r_k, characterized by a curvature decrease from k_{max} to zero.



(**a**) (**b**)

Figure 1. Curvature variation along the path between two straight segments. (**a**) Trapezoid construction: the oblique edges represent the spirals, and the minor base is the circular arc. (**b**) Path transition curve: dot lines represent the spiral-only trajectory without curvature constraint, solid lines are for the curve made of three pieces: two spirals and an arc of circumference.

Procedure 1. Flight path transition curve between two directions

- STEP 1. Firstly, compute the angles ψ_j and ψ_k between an arbitrary axis and r_j and r_k, respectively.
- STEP 2. Assuming $\Delta\psi = |\psi_k - \psi_j|$, κ_{max} the maximum curvature and σ_{max} the maximum sharpness, it is possible to compute $\Delta s = 2\kappa_{max}/\sigma_{max}$, as the length of a virtual curve with maximum sharpness and maximum curvature. It is worth noting that it is called virtual because the heading change constraint is not considered yet.

- STEP 3. The area of the trapezium with major base Δs, minor base l as the length of the circular arc, and height κ_{max} must be equal to $\Delta\psi$. The minor base can be computed as

$$l = \frac{\Delta\psi}{\kappa_{max}} - \frac{\Delta s}{2}$$

If $l > 0$, the path includes a circular arc with curvature κ_{max}. If $l = 0$, then the path includes only two clothoids and the maximum curvature κ_{max} is reached at the middle point. If $l < 0$, the path includes only two clothoids that do not reach the maximum curvature.

- STEP 4. Starting from the intersection point $Q = (x_Q, y_Q)$, the path can be computed by integrating the following equations as follows:

$$\begin{cases} \dot{x}(s) = V\cos\psi(s) \\ \dot{y}(s) = V\sin\psi(s) \\ \dot{\psi}(s) = V\sigma(s) \end{cases} \tag{9}$$

  - For the half-circular arc, the initial conditions are $x(0) = x_Q, y(0) = y_Q$, $\dot{\psi}(0) = V\kappa_{max}$, while $\sigma(s) = 0$ and $s \in \left[0, \frac{l}{2}\right]$.
  - For the spiral curve, in the case of $l > 0$ ($l \leq 0$), the initial conditions are $x(0) = x_Q + x(l/2), y(0) = y_Q + y(l/2)$, $\dot{\psi}(0) = V\kappa_{max}$ ($x(0) = x_Q, y(0) = y_Q$, $\dot{\psi}(0) = V\sqrt{\Delta\psi \cdot \sigma_{max}}$), while $\sigma(s) = -\sigma_{max}$ and $s \in \left[\frac{l}{2}, \frac{\Delta s - l}{2}\right]$.

  These segments represent the second half of the overall curve. The first part can be computed by mirroring the results with respect to the median line between the considered directions. The set of Equation (9) has been employed to mitigate non-linearities in the calculation of heading variation $\dot{\psi}$. This implies that, in the simulation of aircraft motion, the variation law of the bank angle $\dot{\phi}$ can be computed as follows:

$$\begin{aligned} \dot{\phi} &= \sigma\frac{V^2}{g}\cos^2\phi \\ \phi &= \arctan\left(\frac{V^2}{g}\kappa\right) \end{aligned} \tag{10}$$

- STEP 5. The curve must be moved in order to be tangent to both the assigned directions.

3. Path-Planning Algorithm

Given a starting point A with a prescribed initial direction r_A, a target point B, with a prescribed final direction r_B, and a set of N_o obstacles, the optimal flyable path is a smoothed curve, with continuous curvature $\kappa(s)$, composed of segments and flight path transition curves that does not cross any obstacle, passing through several waypoints P_i in the free space.

Its construction involves the use of an optimization algorithm to find the optimal sequence of waypoints P_i. In this paper, a heuristic algorithm Theta* has been modified to generate flyable paths.

The genesis of Theta* can be attributed to A. Nash [35], who originally introduced it as an extension of the A* algorithm. A* is a path-planning approach, primarily preferred for its effectiveness and simplicity of implementation. However, it imposes a limitation on the heading variation, which is typically constrained to multiples of 45°. This restriction, due to the grid-based movement, often results in a path deviating from the true shortest route with several changes in the heading of the vehicle.

Theta* emerged to overcome the A* constraints, particularly the grid-dependent path limitation governing heading variations. It explores the search space similarly to A*, but it optimizes the path during exploration to avoid unnecessary changes in the trajectory

direction. However, flight paths resulting from both A* and Theta* appear as piecewise linear trajectories that are not compliant with the aircraft flight mechanics constraints.

The proposed procedure avoids the need for smoothing in the post-processing phase, allowing the aircraft performance constraints to be taken into account directly in the trajectory optimization process. While in its classical formulation, the arc connecting two points, P_i and P_j, is a straight segment, the proposed solution considers a flyable edge $\Gamma(P_i, P_j)$ with continuous curvature $\kappa(s)$, made of three parts: an initial straight segment starting in P_i, a flight path transition curve with a starting direction r_i, and a segment connecting the ending point of the transition curve and the node P_j, with a direction r_j that must be computed in order to minimize the overall length.

The procedure to connect the node P_i with direction r_i and the node P_j is the following.

Procedure 2. Flyable edge between two points P_i, with direction r_i, and P_j.

- STEP 0—Set an initial guess for the building distance d_c.
- STEP 1—Consider a point $Q^0 = (x_{Q^0}, y_{Q^0})$ to be used as the intersection point between the direction r_i and the candidate direction r_j^0 such that

$$\begin{cases} x_{Q^0} = x_{P_i} + d_c \cos\psi_i \\ y_{Q^0} = y_{P_i} + d_c \sin\psi_i \end{cases} \tag{11}$$

  where ψ_i is the heading angle to describe the direction r_i, and x_{P_i} and y_{P_i} are the coordinates of the point P_i (see Figure 2a).
- STEP 2—Compute r_j^0 as the direction $P_j - Q^0$.
- STEP 3—Build the transition curve using Procedure 1.
- STEP 4—Add two segments: the former connecting P_i with the initial point of the transition curve and the latter connecting the ending point of the transition curve with P_j.
- STEP 5—Compute ΔL the length of the first segment of the just-built flyable path between P_i and P_j.
- STEP 6—If $\Delta L > 0$, then $d_c = d_c - \Delta L$ and return to STEP 1 (see Figure 2b, otherwise return the obtained flyable path $\Gamma(P_i, P_j)$ and the final direction $r_j = r_j^0$.

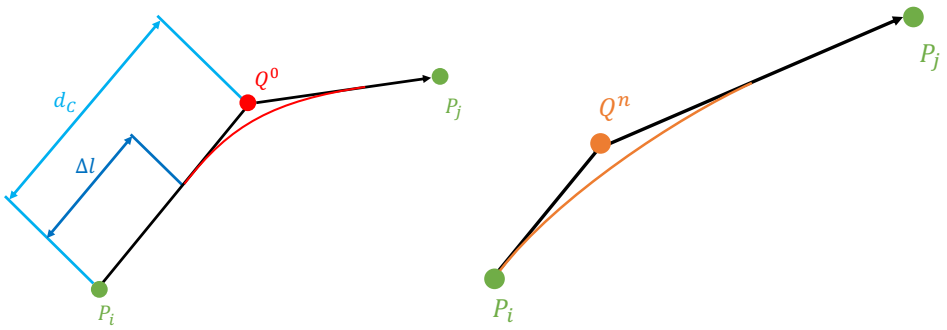
In the original Theta* algorithm, exploration of the space and grid construction occur simultaneously. However, the resolution at which the search space is decomposed becomes crucial for finding the optimal trajectory, especially in obstacle-dense scenarios. In simple scenarios, a high-resolution decomposition would only waste computational resources, while a low-resolution grid in complex environments can result in sub-optimal trajectories. In the proposed algorithm, a variable distance of exploration has been employed, depending on the relative distance from the obstacles.

To speed up the exploration process and avoid the computation of the distances from the obstacles at each step, an image-based solution has been adopted. In particular, a binary representation of the scenario is considered, where the zones occupied by obstacles are marked with ones and the free spaces with zeros. Then, a convolutional filter is applied to obtain a blurred image, as shown in Figure 3b, where the color scale represents the considered exploration distance d_{search}. The blue areas indicate low-resolution zones because they are far from the obstacles, while the green ones stand for high-resolution regions because they are close to the obstacles.

To further reduce the computational time, given a node P_i, a vision cone $\mathbb{V}_i$ of angle η is adopted to limit exploration only to the nodes P_j falling within $\mathbb{V}_i$, with an exploration distance $d_{search}(P_i)$. Denote with $\mathcal{V}_i$ the set of neighbors P_j in the free space, within the vision cone $\mathbb{V}_i$, with a distance $d_{search}(P_i)$ from P_i:

$$\mathcal{V}_i = \{P_j \in V_i : \|P_j - P_i\|_2 = d_{search}(P_i)\}$$

where $\|\cdot\|_2$ indicates the Euclidean norm.



(**a**) (**b**)

Figure 2. The optimal path between two graph nodes is found with an iterative method. (**a**) First step of the method. (**b**) Final result.

The procedure to find the path between the starting point A and the target point B is the following.

Procedure 3. Theta* exploration process to find the flyable path between two points A and B, with assigned directions r_A and r_B, respectively.

- STEP 0—Consider the set $\mathcal{C}$ of explored nodes and the set $\mathcal{O}$ of unexplored nodes consisting of tuples $< P_i, r_i, P_i^h, \mathfrak{F}_i >$, where P_i is the considered node, r_i is the departing direction, P_i^h is the parent node, and $\mathfrak{F}_i$ is the global cost:

$$\mathfrak{F}_i = \mathfrak{H}(A, P_i^h) + L(P_i^h, P_i) + \|B - P_i\|_2$$

  with $\mathfrak{H}(A, P_i^h)$ the length of the path used to fly between A and P_i^h, $L(P_i^h, P_i)$ the length of the flyable edge $\Gamma(P_i^h, P_i)$ between P_i^h and P_i, and $\|B - P_i\|_2$ the Euclidean distance between B and P_i.
  Initialize $\mathcal{C} = \emptyset$ and $\mathcal{O} =< A, r_A, A, \mathfrak{F}_A >$.
- STEP 1—Select and remove from $\mathcal{O}$ the tuple $< P_i, r_i, P_i^h, \mathfrak{F}_i >$ with minimum global cost. Add the tuple to $\mathcal{C}$. If $P_i = B$, then terminate the procedure.
- STEP 2—Compute the vision cone $\mathbb{V}_i$ and build the set $\mathcal{V}_i$. If B is in $\mathbb{V}_i$, add B to the set of neighbors $\mathcal{V}_i$.
- STEP 3—For each node $P_j \in \mathcal{V}_i$, build the flyable paths $\Gamma(P_i, P_j)$ and $\Gamma(P_i^h, P_j)$.
  If $P_j = B$, then two flyable edges are needed to find a trajectory between two points with assigned initial and final directions. In particular, said $\tilde{B} = \frac{P_i + B}{2}$ an intermediate point, two flyable edges $\Gamma(P_i, \tilde{B})$ and $\Gamma(\tilde{B}, B)$ must be computed.

$$\Gamma(P_i, B) = \Gamma(P_i, \tilde{B}) \cup \Gamma(\tilde{B}, B)$$

- STEP 4—For each node P_j, compute the two possible global costs associated to P_j:

$$F_j^1 = \begin{cases} \mathfrak{H}(A, P_i^h) + L(P_i^h, P_j) + \|B - P_j\|_2 & \text{if } \Gamma(P_i^h, P_j) \text{ is feasible} \\ +\infty & \text{otherwise} \end{cases}$$

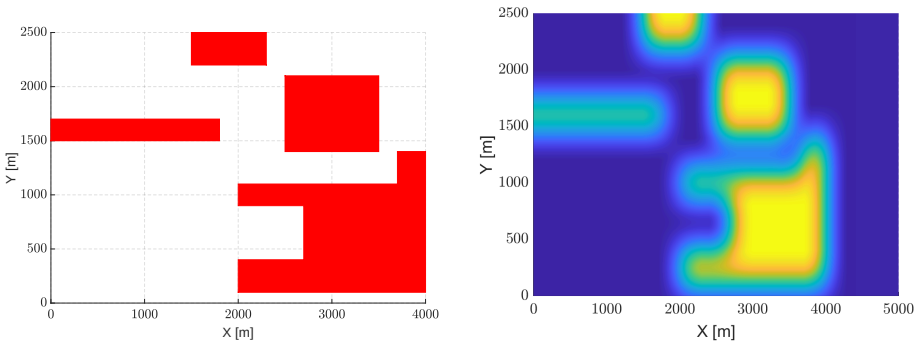
$$F_j^2 = \begin{cases} \mathfrak{H}(A, P_i) + L(P_i, P_j) + \|B - P_j\|_2 & \text{if } \Gamma(P_i, P_j) \text{ is feasible} \\ +\infty & \text{otherwise} \end{cases}$$

  where the edge $\Gamma(\cdot, \cdot)$ is feasible if it does not intersect any obstacle.
- STEP 5—For each node P_j assign to $\mathfrak{F}_j$ the minimum global cost associated to P_j and the relative parent node

$$P_j^h = \begin{cases} P_i & \text{if } F_j^2 < F_j^1 \\ P_i^h & \text{otherwise} \end{cases}$$

- STEP 6—Consider the tuple $< P_j, r_j, P_j^h, \mathfrak{F}_j >$. If $\mathcal{C}$ contains a tuple $< P_j, r_j^0, P_j^{h^0}, \mathfrak{F}_j^0 >$, if $\mathfrak{F}_j^0 > \mathfrak{F}_j$ then remove $< P_j, r_j^0, P_j^{h^0}, \mathfrak{F}_j^0 >$ and add $< P_j, r_j, P_j^h, \mathfrak{F}_j >$ to $\mathcal{C}$. If $\mathcal{C}$ does not contain any tuple with P_j, then add $< P_j, r_j, P_j^h, \mathfrak{F}_j$ to $\mathcal{C}$.
- STEP 7—Go to STEP 1.

To reconstruct the optimal flyable path between A and B, find the tuple with B in the set $\mathcal{C}$ and move to its parent node. Iterate the process until the parent node is A.

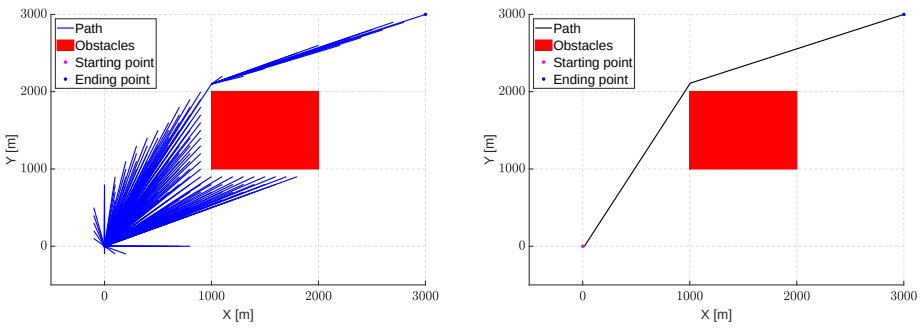


(**a**) (**b**)

Figure 3. Image-based solution adopted to speed up the process. The distance between two nodes of the grid depends on their distances from the obstacles. (**a**) Example of binary representation of the scenario. (**b**) Example of filtered scenario; the blue areas indicate the low-resolution zones because they are far from the obstacles while the green ones stand for the high-resolution regions because they are close to the obstacles.

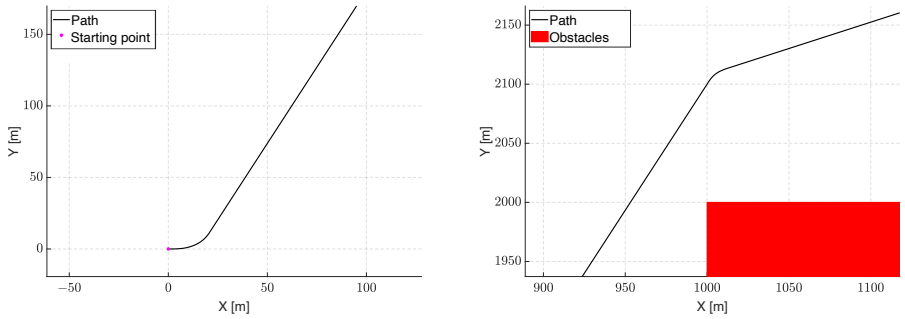
To preliminarily evaluate the algorithm capability in identifying feasible paths, a simple example is presented in Figure 4, where the initial point and the target point A and B are given, with starting and target directions ψ_{start} and ψ_{end}, respectively. The scenario presents one convex obstacle to be avoided at the center of the environment. In Figure 4b, the final path avoiding the obstacle is shown, while in Figure 4a, the blue lines show the arcs explored by the algorithm during the search. The results are summarized in Table 1.

In Figure 5, the details of the planned path for this scenario are shown, highlighting the starting turn maneuver, which is needed to deviate from the preassigned initial slope, and the turn needed to go around the obstacle.



(**a**) (**b**)

Figure 4. Test case #0: Scenario with one obstacle, represented in red, starting point at $A = (0, 0)$ m and target point at $B = (3000, 3000)$ m. (**a**) Paths (blue lines) explored by the algorithm during the optimization process. (**b**) Final path (black line).



(**a**) (**b**)

Figure 5. Test case #0: Details of the optimal flyable path. (**a**) Initial turn maneuver. (**b**) Turn around the obstacle.

Table 1. Test case #0: Results in terms of time, length, and number of explored nodes.

	Value
d_{search} [m]	100
Time [s]	16.61
Path length [m]	4527.4
Explored nodes	190
Total generated nodes	286

4. Results

For a more comprehensive assessment of the algorithm effectiveness, the path planner was tested on five different scenarios. The first test case presents a scenario with several convex and concave obstacles to test the ability of the algorithm to avoid the concavity of obstacles when the vision cone is also in use. In the second test case, the capability of the procedure to use a variable exploration distance is stressed, comparing results with a Theta* without the new feature. The third test case presents a comparison of the results provided by different variants of the algorithm with varying combinations of the proposed features. The fourth test simulates the use of the path planner in a real urban scenario. In particular, a dense area of Napoli (IT) is considered, with the aircraft flying at a fixed flight altitude of $h = 10$ m, avoiding buildings. Finally, the last test case is considered to verify the accurate tracking of an aircraft along the planned trajectory.

As shown in Table 2, the different algorithm variants will be referred to using the following notation: θ^*_C for the base algorithm implementing only the path transition curve as edges; θ^*_V for the algorithm with the vision cone option active; θ^*_G in the case when the variable exploration distance is used; θ^*_F indicates the algorithm using all the optional features, i.e., the vision cone and the variable exploration distance.

Table 2. Algorithm variants. Table of active features: (x) stands for not active, (+) active

	Smooth Transition	**Vision Cone**	**Variable Grid**
θ^*	x	x	x
θ^*_C	+	x	x
θ^*_V	+	+	x
θ^*_G	+	x	+
θ^*_F	+	+	+

4.1. Test Case #1

This simulation test case evaluates the performance of the proposed algorithm, with and without the vision cone option, in terms of path length and computational burden.

The parameters used in the first scenario are listed in Table 3. The scenario is composed of four convex and two concave obstacles. To stress the algorithm and prove its effectiveness, the starting point is placed inside the concave obstacle (Figure 6), and the positions of the obstacles are chosen in order to create narrow passages. The starting and ending directions are arbitrarily chosen.

Figure 6 shows the result for the first test case. In particular, Figure 6a,b represent the explored paths and the final trajectory, respectively, when the vision cone option is not active, while Figure 6c,d show the results of the algorithm with the vision cone option active.

As can be seen, the aircraft is able to fly effectively in the operational scenarios, passing through narrow passages and avoiding becoming stuck in concave obstacles.

During the exploration phase, the use of the vision cone reduces the number of explored nodes, allowing for a lower number of explored paths, as shown in Figure 6a,c. In particular, in this case study, the lower number of tested trajectories results in a final trajectory (Figure 6d) slightly longer than the path obtained without an active vision cone (Figure 6b), but with the advantage of a reduced computational burden, as highlighted in Table 4.

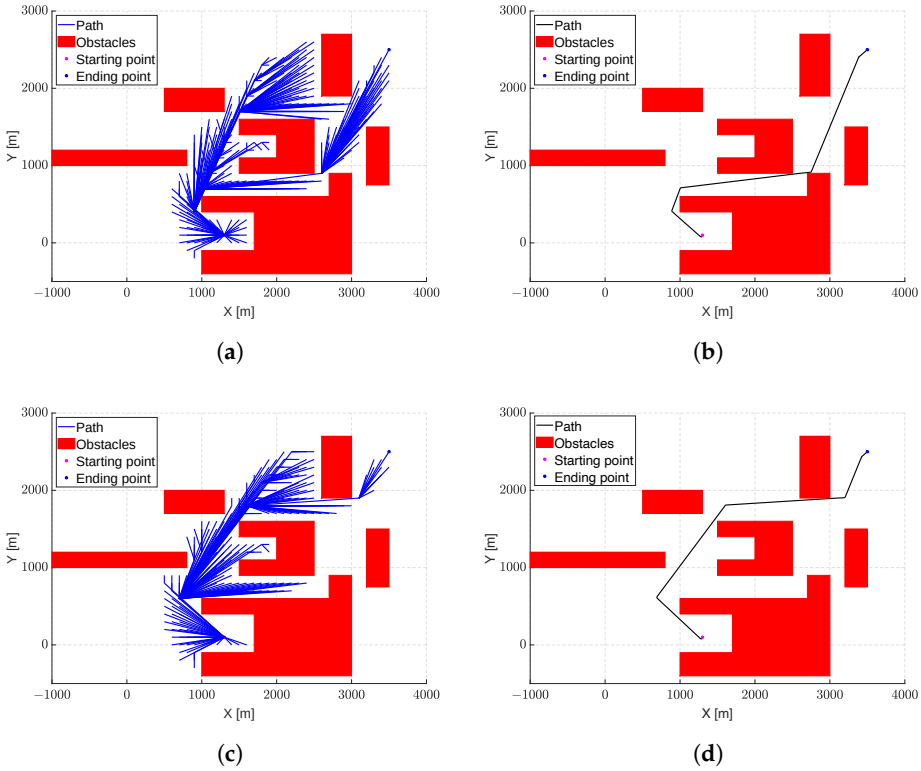


Figure 6. Test case #1. Comparison between θ_C^* and θ_V^* algorithms. Obstacles are represented in red, explored paths with solid blue lines, and final paths with solid black lines. (**a**,**b**) θ_C^* algorithm. (**c**,**d**) θ_V^* algorithm.

Table 3. Test-case #1: scenario parameters.

	Value
$\boldsymbol{A}$ [m]	(1300,100)
$\boldsymbol{\psi_{start}}$ [deg]	270
$\boldsymbol{B}$ [m]	(3500,2500)
$\boldsymbol{\psi_{end}}$ [deg]	50
η [deg]	80

Table 4. Test-case #1. Comparison of time, length, and grid nodes between the algorithm with and without the vision cone.

	$\boldsymbol{\theta_C^*}$	$\boldsymbol{\theta_V^*}$
Time [s]	42.40	13.21
Path length [m]	4389.2	4606.9
Explored nodes	305	245
Total nodes generated	368	294

4.2. Test Case #2

The main aim of the second test case is to assess the algorithm ability to use variable exploration distance to find the shortest path with a reduced computational burden. The second operational scenario consists of three convex obstacles and one concave obstacle, arranged to form narrow corridors along the path from the starting point to the destination point. Table 5 lists the parameters used. The obstacles are placed to highlight the different results obtained by using fixed or variable exploration distances. In particular, Figure 7a,c represent the explored nodes running the algorithm with fixed exploration distances equal to $d_{search} = 50$ m and $d_{search} = 100$ m, respectively, while the final paths obtained at the end of these runs are represented in Figure 7b,d, respectively. In this scenario, having more explored nodes is advantageous in terms of the final trajectory length, as it allows for close passage between obstacles. On the other hand, a lower number of points does not allow for finding a solution of the same quality, as the algorithm avoids the passage through the narrow corridor, leading to a longer trajectory.

A better result can be achieved using a variable exploration distance, as illustrated in Figure 7e,f. In particular, Figure 7e demonstrates the benefits of employing the new feature, creating a region with a higher density of nodes only in the neighbors of obstacles. This allows the identification of a final trajectory (Figure 7f) very similar to the one obtained with more nodes, as shown in Figure 7b. However, this result is achieved with fewer generated nodes and, consequently, with a lower computational burden.

Table 6 shows the comparison between the proposed algorithms in terms of computational times and trajectory lengths. As can be observed, the algorithm featuring a variable search distance ensures a good solution in terms of final trajectory length with a lower computational burden compared to the other two algorithm variants using a fixed exploration distance.

Table 5. Test-case #2: scenario parameters.

	Value
$\boldsymbol{A}$ [m]	(1800, 500)
$\boldsymbol{\psi_{start}}$ [deg]	0
B [m]	(5000, 1500)
$\boldsymbol{\psi_{end}}$ [deg]	90

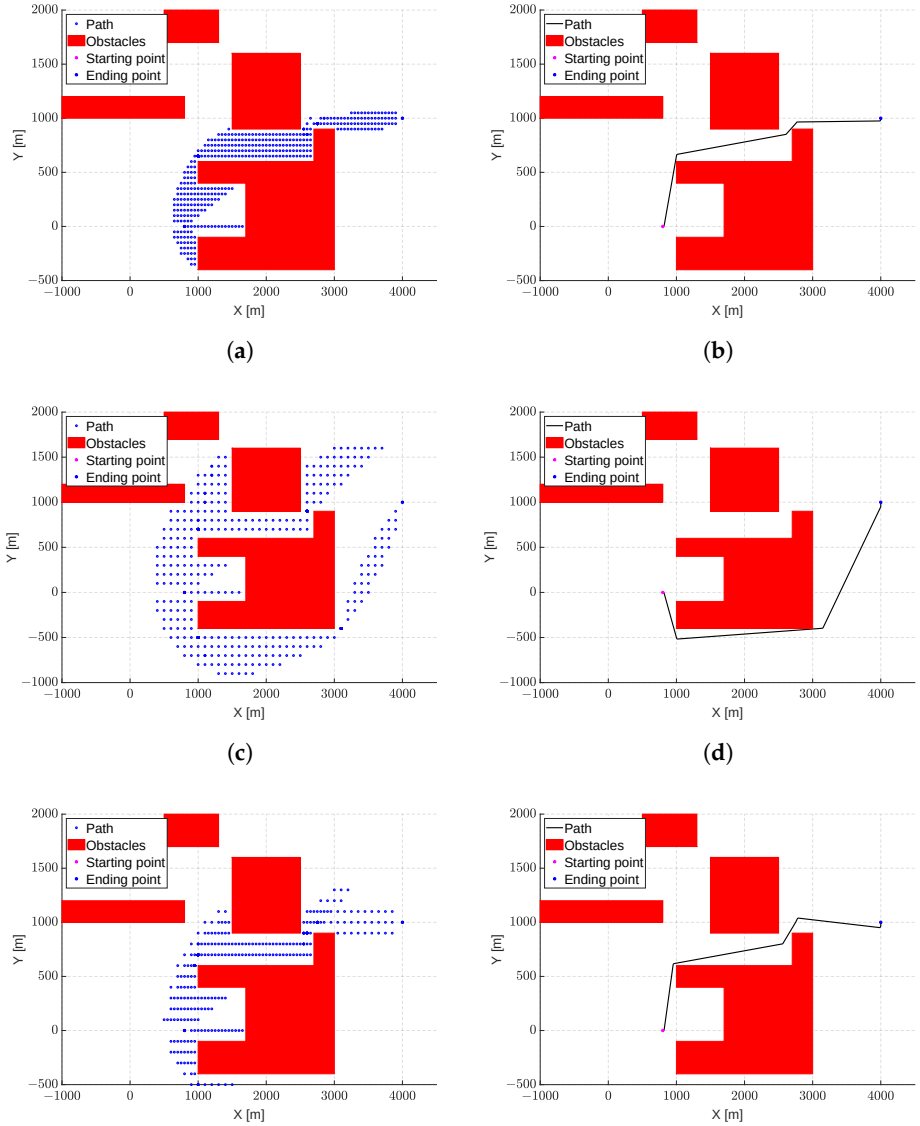


Figure 7. Test case #2. Comparison between θ^*_C and θ^*_G algorithms. Obstacles are represented in red, explored nodes with blue dots, and final paths with solid black lines. (**a**,**b**) θ^*_C algorithm with fixed search distance $d_{search} = 50$ m. (**c**,**d**) θ^*_C algorithm with fixed search distance $d_{search} = 100$ m. (**e**,**f**) θ^*_G algorithm.

Table 6. Test-case #2. Comparison of time, length, and grid nodes between the algorithm with fixed grid and search distance 50 m; fixed grid and search distance 100 m; variable grid.

	θ^*_C	θ^*_C	θ^*_G
d_{search} [m]	50	100	variable
Time [s]	41.59	32.88	22.92
Path length [m]	3761.2	4348.1	3845.5
Explored nodes	336	299	285
Total generated nodes	465	418	450

4.3. Test Case #3

This scenario, consisting of three convex obstacles and one concave obstacle, allows us to test whether the proposed algorithm is able to reduce the computational burden compared to its variants, which use either the vision cone or the variable exploration distance.

The specific arrangement of obstacles forces the algorithms to construct similar paths, allowing us to evaluate the solution goodness in terms of computational burden rather than path length.

The parameters used for this simulation are stated in Table 7.

Figure 8 shows the trajectories explored with the algorithms, while Figure 9 illustrates the final paths, which appear very similar in all the analyzed cases.

Table 8 presents the results in terms of computation time and trajectory lengths. As can be seen, all the algorithm variants are able to identify trajectories of similar lengths. However, the θ_F^* algorithm, which features both the vision cone and the variable exploration distance, provides not only the shortest trajectory but, above all, shows a significant reduction in the computational time compared to the other algorithm variants.

Table 7. Test-case #3: scenario parameters.

	Value
$\boldsymbol{A}$ [m]	(1800, 500)
$\boldsymbol{\psi_{start}}$ [deg]	0
$\boldsymbol{B}$ [m]	(3500, 3000)
$\boldsymbol{\psi_{end}}$ [deg]	0

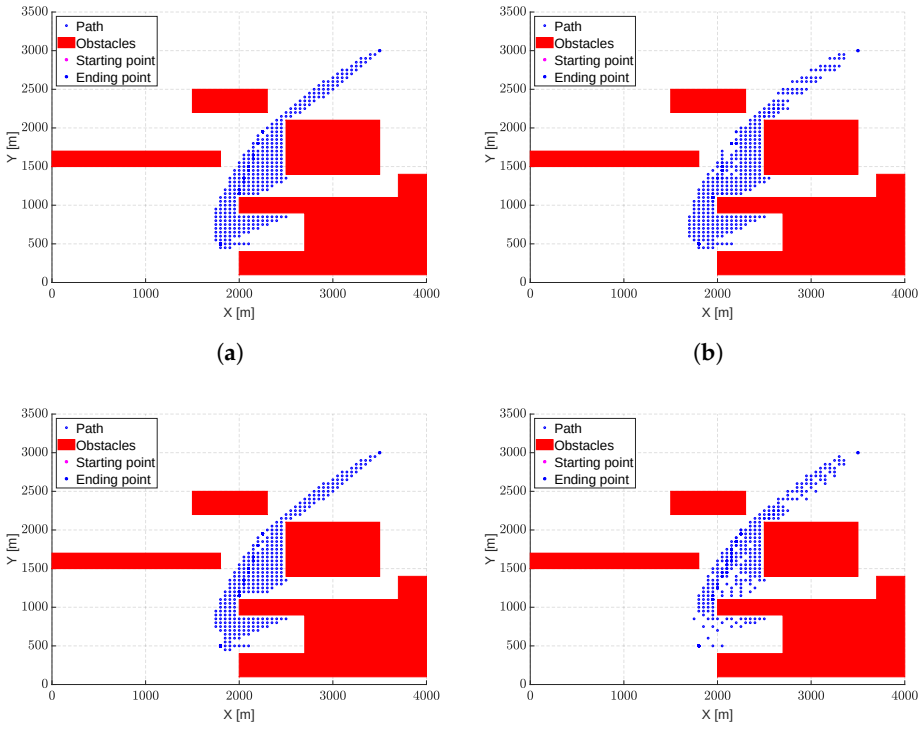


Figure 8. Test case #3. Comparison between θ_C^*, θ_G^*, θ_V^*, and θ_F^* algorithms. Obstacles are represented in red, and explored paths with solid blue lines. (**a**) Explored paths with θ_C^* algorithm. (**b**) Explored paths with θ_G^* algorithm. (**c**) Explored paths with θ_V^* algorithm. (**d**) Explored paths with θ_F^* algorithm.

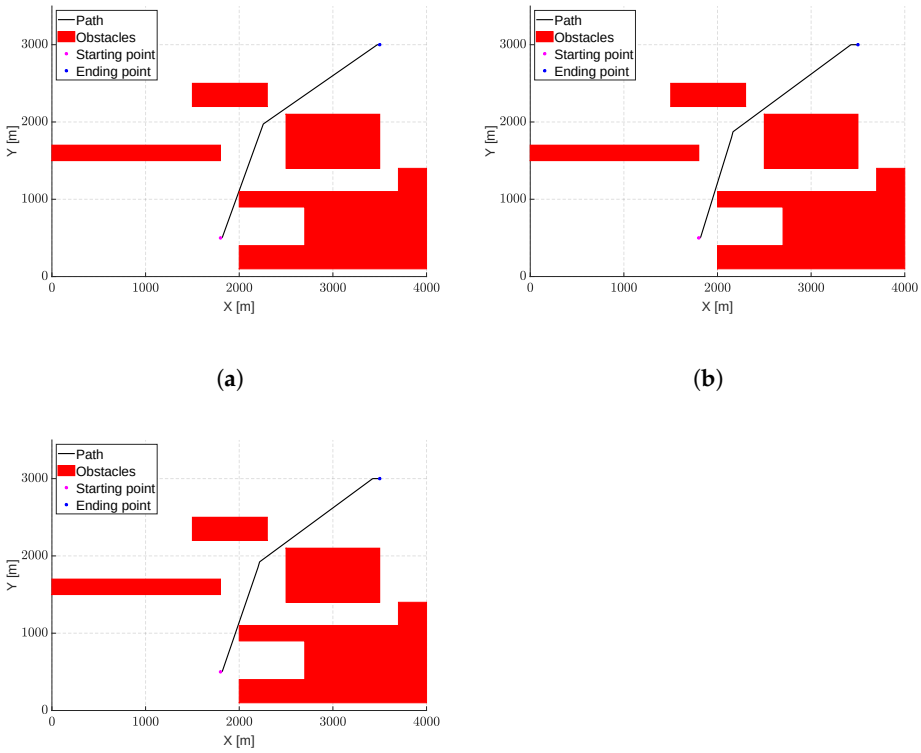


Figure 9. Test case #3. Final paths. Obstacles are represented in red, and final paths with solid black line. (**a**) Final path with θ_C^* and θ_V^* algorithms. (**b**) Final path with θ_G^* algorithm. (**c**) Final path with θ_F^* algorithm.

Table 8. Test-case #3. Comparison of time, length, and grid nodes between the algorithm with Theta* clothoid; Theta* clothoid with variable grid; Theta* clothoid with vision cone; Theta* clothoid with both variable grid and vision cone.

	θ_C^*	θ_V^*	θ_G^*	θ_F^*
d_{search} [m]	50	50	variable	variable
Time [s]	35.39	15.36	33.94	11.80
Path length [m]	3168.1	3168.1	3194.0	3183.5
Explored nodes	329	316	343	258
Total generated nodes	447	386	624	384

4.4. Test Case #4: Urban Scenario

The aim of test case #4 is to evaluate the algorithm's capability of identifying the optimum path in a real urban environment. The parameters used in this scenario are shown in Table 9. The coordinates are given in terms of longitude and latitude; in particular, this scenario replicates the area of Naples city located between Piazza Nazionale and Centro Direzionale.

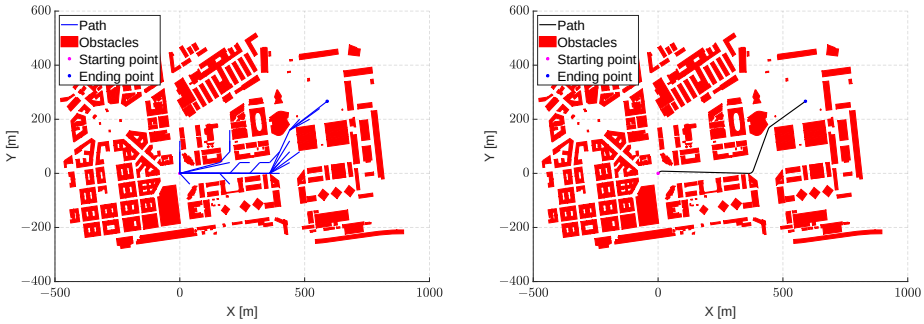
The path search is carried out with a fixed exploration distance $d_{search} = 40$ m. This distance turned out to be a fair compromise between two needs: having a higher number of points to navigate narrow passages and limiting the computational burden even in a scenario full of obstacles.

Figure 10 represents the results for this urban scenario. Figure 10a shows the exploration phase, where the algorithm performs an extensive search of its surroundings to find a feasible path compliant with the UAV's performance. The final path identified is presented in Figure 10b.

Table 10 summarizes the obtained results. Due to the large number of obstacles involved in this scenario, the computational burden appears significantly higher compared to the previous test cases. A time reduction could be expected through a pre-processing work aimed at limiting the number of obstacles considered in each iteration, but this goes beyond the scope of this test.

Table 9. Test-case #4: scenario parameters.

	Value
$\boldsymbol{A}$ (lon,lat) [deg, deg]	(14.277, 40.8566)
$\boldsymbol{\psi_{start}}$ [deg]	45
$\boldsymbol{B}$ (lon,lat) [deg, deg]	(14.284, 40.859)
$\boldsymbol{\psi_{end}}$ [deg]	45



(**a**) (**b**)

Figure 10. Test case #4. Simulation results for the search of the path between the starting and the target point in a realistic urban scenario in Naples. Obstacles are represented in red, explored paths with solid blue lines, and final paths with solid black lines. (**a**) Exploration phase. (**b**) Final path.

Table 10. Test-case #4. Results in terms of time, length and grid nodes of the algorithm for an urban scenario.

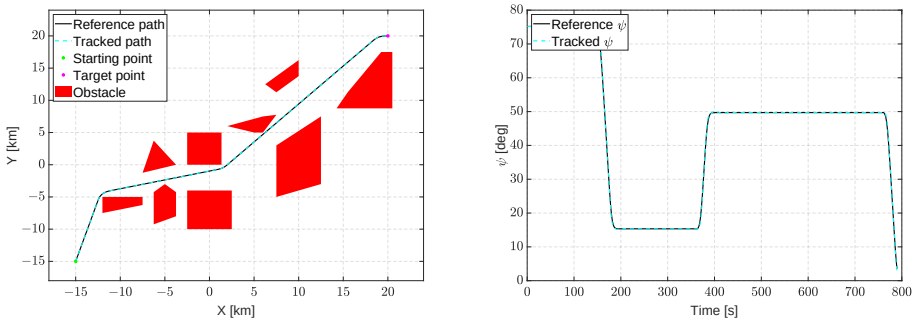
	Value
$d_{search}\,[m]$	40
Time [s]	148.91
Path length [m]	734.04
Explored nodes	37
Total generated nodes	63
Number of obstacles	336

4.5. Test Case #5: Simulation of a Short Cruise Path

In this test case, the problem of planning an optimal path for guiding an aircraft during the cruise phase to avoid specific no-fly zones is considered. The scenario involves two obstacles, a starting point $A = [-15, -15]$ km, a target point $B = [20, 20]$ km, an initial heading $\psi = 75$ deg, and a final heading $\psi = 0$ deg. To ensure realism and reproducibility, a general aviation aircraft with characteristics outlined in [59] was considered for testing. For trajectory tracking, a control system based on classical control methods [60] was developed.

After planning the optimal path with θ_F^*, considering $k_{max} = 6 \times 10^{-4} \frac{1}{m}$ and $\sigma_{max} = 8.2 \times 10^{-5} \frac{1}{m \cdot s}$, trajectory tracking was simulated to verify its actual feasibility at the cruise speed of $V = 67 \frac{m}{s}$. Figure 11a displays the planned and tracked path, while Figure 11b illustrates the desired and actual heading of the aircraft. As observed, the aircraft effectively follows the trajectory, and the only minor discrepancy arises from the sharpness

discontinuity, resulting in a jump in the roll angular velocity. Nevertheless, such a model error translates into a tracking error maximum of 7 meters.



(**a**) (**b**)

Figure 11. Test-case #5. Simulation results. (**a**) Planned path in black solid line and tracked trajectory in dashed cyan. (**b**) Reference and tracked heading during flight.

5. Discussion

The algorithm's efficacy was comprehensively evaluated through four distinct scenarios.

In scenarios where the vision cone was active, the algorithm demonstrates a reduction in the number of explored nodes, leading to a more efficient exploration phase. However, such a feature occasionally results in slightly longer trajectories. Furthermore, as highlighted in the first test case, featuring convex and concave obstacles, the algorithm is able to overcome concave structures while using the vision cone. This capability is crucial for scenarios with intricate obstacles where maintaining awareness of the environment is pivotal.

The implementation of the variable exploration distance showcased its advantages, particularly in scenarios with narrow passages. By dynamically adjusting node density, the algorithm achieved optimal trajectories with a reduced computational cost compared to standard implementations. The results were emphasized in the second test case. A comparison with a Theta* employing constant distances demonstrated the algorithm's superior performance when dynamically adjusting node densities.

Across all scenarios, the algorithm consistently performed well in terms of trajectory lengths. The θ_F^* variant, incorporating both the vision cone and variable exploration distance, emerged as the most efficient in terms of both trajectory length and computational time.

In particular, the third test case also involved a comprehensive comparison of algorithm variants, each incorporating different combinations of optional features. Notably, the use of both vision cone and variable exploration distance features (denoted as θ_F^*) resulted in better trajectories in terms of overall length with a significant reduction in computational burden. This indicates the synergistic benefits of employing multiple features.

The final test case simulated the algorithm performance in a real urban environment in Naples, Italy. The algorithm successfully navigated a dense urban area, avoiding obstacles and producing feasible trajectories. While the computational burden increased due to the complexity of the scenario, the ability to find suitable trajectories in cluttered environments while managing computational resources is crucial for real-world UAV applications.

To highlight the effectiveness of the proposed methodology in terms of the ability to find the minimum-length path and avoid collisions with environmental obstacles, this section presents a comparison with three algorithms with the same objectives: the Essential Visibility Graph (EVG) with Dubins-based smoothing [56,61], a custom graph-based solution with clothoids [17], and an RRT-based algorithm, named the Stack-RRT* algorithm [62].

The comparison between the EVG and the proposed algorithm is made by considering four different configurations of the same scenario, whose parameters are summarized in Table 11. As depicted in Table 12, the EVG is faster and able to guarantee a proved

optimal path in terms of length. However, as shown in Figure 12, the resulting trajectory is composed of several straight segments, requiring a post-processing smoothing procedure, implemented using Dubins arcs, to make it easily flyable by a fixed-wing aircraft. This approach leads to the presence of discontinuities in curvature $\kappa(s)$. The proposed θ_F^* algorithm is able to create a completely flyable path that can be easily followed by the aircraft, with a slight increase in the total length (maximum 1.6%). Regarding the computation time, there is a clear distinction between the processing times required by the proposed algorithm compared to those presented in [61]. Undoubtedly, the lack of optimization in the code, as well as the implementation in Matlab, has an impact on such extended processing times. The computer considered in this analysis is a laptop with an i5-8250U processor and 8GB of RAM. Certainly, the number of nodes explored in Theta* is much higher than those analyzed in an Essential Visibility Graph. From this perspective, the EVG may appear superior, but in the presence of uncertainty and scenarios involving moving obstacles, Theta* can adapt more easily, even from a procedural standpoint.

The comparison with another clothoid-based approach [17] is faced using the scenario of the test case #3 (see Section 4.3). The parameters are summarized in Table 7. The paths for the comparison are illustrated in Figure 13, while the results are shown in Table 13. It is worth noting that, although both models provide easily flyable trajectories for fixed-wing aircraft, the proposed algorithm appears to be more efficient, ensuring a shorter trajectory at a lower computational burden.

Finally, a comparison with an RRT-based algorithm is presented, considering four different scenarios introduced in [62]. The results, in terms of generated paths, are depicted in Figure 14. As shown in the figures, θ_F^* is capable of planning paths similar to Stack-RRT*, effectively addressing challenges posed by non-convex obstacles. Additionally, as emphasized in Table 14, θ_F^* can identify paths with lengths comparable to the outcomes reported in [62]. Regarding computation times, although they may appear high, it is important to consider both the type of processor available and the lack of code optimization.

Table 11. Comparison θ_F^*—EVG: scenario parameters.

	Value
$\boldsymbol{A}$ [m]	(20,000, 20,000)
$\boldsymbol{\psi_{start}}$ [deg]	225
$\boldsymbol{B_{(a)}}$ [m]	(−15,000, −15,000)
$\boldsymbol{B_{(b)}}$ [m]	(−10,000, 0)
$\boldsymbol{B_{(c)}}$ [m]	(0, 0)
$\boldsymbol{B_{(d)}}$ [m]	(0, −15,000)
$\boldsymbol{\psi_{end}}$ [deg]	180

Table 12. Comparison θ_F^*—EVG. Comparison of the algorithms in terms of time and path length.

	(a)	**(b)**	**(c)**	**(d)**
Time $\boldsymbol{\theta_F^*}$ [s]	85.98	122.81	45.40	71.21
Time **EVG** [s]	1.26	1.52	0.967	1.48
Path length $\boldsymbol{\theta_F^*}$ [m]	51,595.9	38,782.1	29,616.0	41,846.7
Path length **EVG** [m]	51,402.1	38,169.0	29,298.4	41,597.1

Table 13. Comparison θ_F^*—clothoid-based model. Comparison of the algorithms in terms of time and path length.

	$\boldsymbol{\theta_F^*}$	**Clothoid-Based Model**
Time [s]	11.50	16.83
Path length [m]	5273.0	5754.1

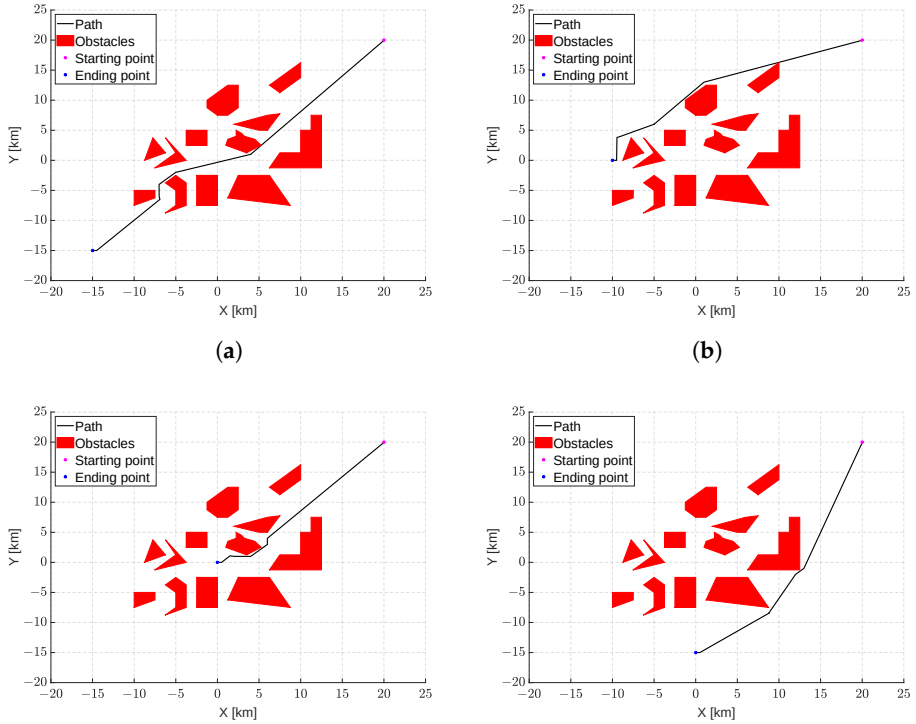


Figure 12. Comparison θ_F^*—EVG. Final paths obtained with θ_F^* algorithm for the scenario proposed in [61]. Obstacles are represented in red and final paths with solid black lines. (**a**) Explored paths with θ_F^* algorithm and ending point $\boldsymbol{B}_{(a)}$. (**b**) Explored paths with θ_F^* algorithm and ending point $\boldsymbol{B}_{(b)}$. (**c**) Explored paths with θ_F^* algorithm and ending point $\boldsymbol{B}_{(c)}$. (**d**) Explored paths with θ_F^* algorithm and ending point $\boldsymbol{B}_{(d)}$.

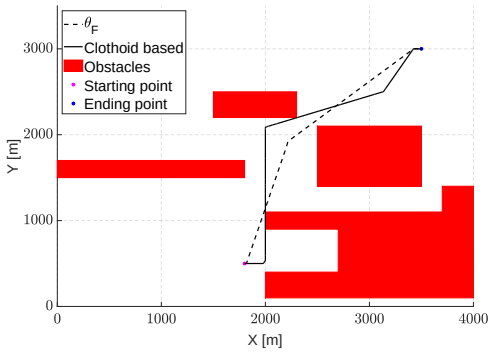


Figure 13. Comparison θ_F^*—clothoid-based solution. Final path obtained with θ_F^* (dotted line) and clothoid-based algorithm (solid line) for the scenario proposed in Section 4.3. Obstacles are represented in red and final paths with solid black lines.

Table 14. Results of θ_F^* for the scenarios presented in Figure 14. Results in terms of time and path length.

	(a)	(b)	(c)	(d)
Time [s]	7.82	4.68	6.90	21.35
Path length [m]	81.73	74.73	46.30	123.99
Explored nodes	93	49	19	136
Total generated nodes	143	86	49	206

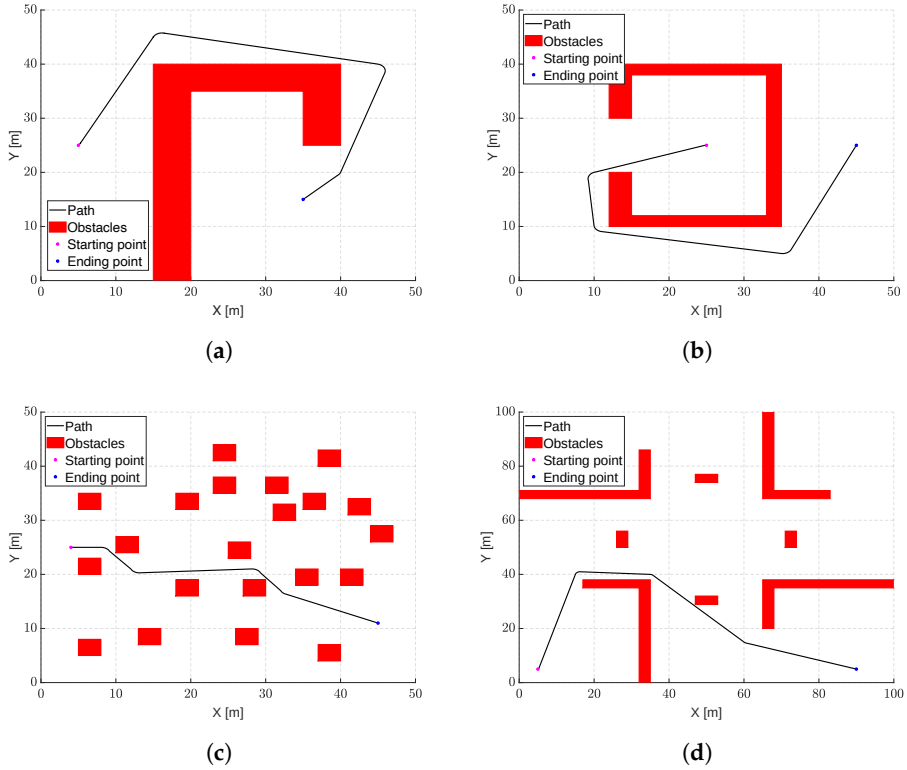


Figure 14. Comparison θ_F^* - Stack-RRT*. Final paths obtained with θ_F^* algorithm for the scenario proposed in [62]. Obstacles are represented in red and final paths with solid black lines. (**a**) Explored paths with θ_F^* algorithm for the scenario presented in Figure 5 of Ref. [62]. (**b**) Explored paths with θ_F^* algorithm for the scenario presented in Figure 6 of Ref. [62]. (**c**) Explored paths with θ_F^* algorithm for the scenario presented in Figure 7 of Ref. [62]. (**d**) Explored paths with θ_F^* algorithm for the scenario presented in Figure 8 of Ref. [62].

6. Conclusions

In this paper, a path-planning strategy for fixed-wing UAVs is proposed by enhancing the so-called Theta* algorithm. The path design involves a piece-wise curve composed of circular and spiral arcs, ensuring compliance with aircraft performance constraints. In particular, the transition between the initial and the target directions is made by means of Euler spiral-based transition curves, comprising two spirals and one circular arc having a curvature compliant with the specific aircraft characteristics.

The optimization of the final path is based on an enhanced version of the Theta* algorithm, where arcs between nodes are defined by the aforementioned path transition curves. This solution is able to generate trajectories without discontinuities, ensuring that the entire flight path is compliant with the performance of the aircraft. To reduce the algorithm's burden on the CPU, the computational grid of Theta* is dynamically generated with a variable step size, taking into account actual distances to obstacles. The exploration of the search space is further enhanced by the introduction of a vision cone, limiting the algorithm exploration only to relevant areas and preventing unnecessary maneuvers.

The results of comprehensive tests on five distinct scenarios demonstrate the effectiveness of the proposed algorithm. The first test, involving scenarios with both convex and concave obstacles, highlights the capability of the algorithm to find feasible paths in complex environments, also with the use of the vision cone feature, which is mainly aimed at reducing computational time. The second test emphasizes the advantages of adopting a variable exploration distance, showing its superiority over a standard Theta* algorithm. In the third test, a comparative analysis between the algorithm variants, incorporating all the developed features, demonstrates the robustness of the proposed approach. The fourth test simulates a real urban scenario in Naples, Italy, where the UAV, flying at a fixed altitude,

successfully avoids buildings while reaching the destination point. Finally, the last test case verifies the accurate tracking along the planned trajectory using a high-fidelity simulator.

In summary, the results show that the proposed path-planning procedure allows the aircraft to effectively navigate through narrow passages and avoid concave obstacles. During the exploration phase, the use of a vision cone significantly reduces the number of nodes explored, resulting in a slightly longer final trajectory but improving computational efficiency.

Furthermore, the adoption of a variable exploration distance, creating a region of higher node density only in the proximity of obstacles, improves the algorithm's effectiveness by achieving a final path similar to a trajectory obtained with a denser presence of nodes but with a reduced computational burden.

Further improvement opportunities exist, especially in scenarios with a high number of obstacles. Exploring pre-processing techniques to limit obstacles considered in each iteration could potentially enhance computational efficiency. Furthermore, additional refinements and applications can be explored, considering 3D scenarios and the use of path transition curves including climb maneuvers.

Author Contributions: Conceptualization, E.D. and I.N.; data curation, S.R.B. and G.R.; formal analysis, S.R.B., E.D. and I.N.; investigation, S.R.B.; methodology, E.D., I.N. and G.R.; resources, L.B.; software, G.R.; supervision, E.D.; validation, S.R.B., I.N. and G.R.; writing—original draft, S.R.B., E.D., I.N., G.R. and L.B.; Writing—review and editing, S.R.B., E.D., I.N., G.R. and L.B. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the research project—ID:20222N4C8E "Resilient and Secure Networked Multivehicle Systems in Adversary Environments" granted by the Italian Ministry of University and Research (MUR) within the PRIN 2022 program, funded by the European Union through the PNRR program.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Federal Aviation Administration. *FAA Aerospace Forecast. Fiscal Years 2022–2042*; Federal Aviation Administration: Washington, DC, USA, 2022.
2. Ramesh, P.; Jeyan, J.M.L. Comparative analysis of the impact of operating parameters on military and civil applications of mini unmanned aerial vehicle (UAV). In *Proceedings of the AIP Conference Proceedings*; AIP Publishing LLC: College Park, MD, USA, 2022 ; Volume 2311, p. 030034.
3. Aggarwal, S.; Kumar, N. Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges. *Comput. Commun.* **2020**, *149*, 270–299. [CrossRef]
4. Gul, F.; Mir, I.; Abualigah, L.; Sumari, P.; Forestiero, A. A consolidated review of path planning and optimization techniques: Technical perspectives and future directions. *Electronics* **2021**, *10*, 2250. [CrossRef]
5. D'Amato, E.; Mattei, M.; Notaro, I. Bi-level flight path planning of UAV formations with collision avoidance. *J. Intell. Robot. Syst.* **2019**, *93*, 193–211. [CrossRef]
6. Goerzen, C.; Kong, Z.; Mettler, B. A survey of motion planning algorithms from the perspective of autonomous UAV guidance. *J. Intell. Robot. Syst.* **2010**, *57*, 65–100. [CrossRef]
7. Dasgupta, B.; Gupta, A.; Singla, E. A variational approach to path planning for hyper-redundant manipulators. *Robot. Auton. Syst.* **2009**, *57*, 194–201. [CrossRef]
8. Shukla, A.; Singla, E.; Wahi, P.; Dasgupta, B. A direct variational method for planning monotonically optimal paths for redundant manipulators in constrained workspaces. *Robot. Auton. Syst.* **2013**, *61*, 209–220. [CrossRef]
9. la Cour-Harbo, A.; Bisgaard, M. State-control trajectory generation for helicopter slung load system using optimal control. In *Proceedings of the AIAA Guidance, Navigation, and Control Conference*; AIAA: Chicago, IL, USA, 2009 ; p. 6296.
10. Xu, N.; Kang, W.; Cai, G.; Chen, B.M. Minimum-time trajectory planning for helicopter UAVs using computational dynamic optimization. In Proceedings of the 2012 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Seoul, Republic of Korea, 14–17 October 2012; pp. 2732–2737.

11. D'Amato, E.; Mattei, M.; Notaro, I. Distributed reactive model predictive control for collision avoidance of unmanned aerial vehicles in civil airspace. *J. Intell. Robot. Syst.* **2020**, *97*, 185–203. [CrossRef]
12. Liu, P.; Yu, H.; Cang, S. Geometric analysis-based trajectory planning and control for underactuated capsule systems with viscoelastic property. *Trans. Inst. Meas. Control.* **2017**, *40*, 0142331217708833. [CrossRef]
13. Duan, H.; Zhao, J.; Deng, Y.; Shi, Y.; Ding, X. Dynamic Discrete Pigeon-Inspired Optimization for Multi-UAV Cooperative Search-Attack Mission Planning. *IEEE Trans. Aerosp. Electron. Syst.* **2021**, *57*, 706–720. [CrossRef]
14. Dubins, L.E. On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. *Am. J. Math.* **1957**, *79*, 497–516. [CrossRef]
15. Owen, M.; Beard, R.W.; McLain, T.W. Implementing Dubins airplane paths on fixed-wing uavs. In *Handbook of Unmanned Aerial Vehicles*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 1677–1701.
16. Babel, L. Coordinated target assignment and UAV path planning with timing constraints. *J. Intell. Robot. Syst.* **2019**, *94*, 857–869. [CrossRef]
17. Blasi, L.; D'Amato, E.; Notaro, I.; Raspaolo, G. Clothoid-Based Path Planning for a Formation of Fixed-Wing UAVs. *Electronics* **2023**, *12*, 2204. [CrossRef]
18. Scherer, S.; Singh, S.; Chamberlain, L.; Elgersma, M. Flying fast and low among obstacles: Methodology and experiments. *Int. J. Robot. Res.* **2008**, *27*, 549–574. [CrossRef]
19. Schøler, F.; Cour-Harbo, A.; Bisgaard, M. Configuration space and visibility graph generation from geometric workspaces for uavs. In Proceedings of the AIAA Guidance, Navigation, and Control Conference, Portland, Oregon, 8–11 August 2011.
20. Schøler, F.; la Cour-Harbo, A.; Bisgaard, M. Generating approximative minimum length paths in 3D for UAVs. In Proceedings of the Intelligent Vehicles Symposium (IV), Madrid, Spain, 3–7 June 2012; pp. 229–233.
21. Maini, P.; Sujit, P.B. Path planning for a UAV with kinematic constraints in the presence of polygonal obstacles. In Proceedings of the 2016 International Conference on Unmanned Aircraft Systems (ICUAS), Arlington, VA, USA, 7–10 June 2016; pp. 62–67.
22. Bortoff, S.A. Path planning for UAVs. In Proceedings of the American Control Conference, Chicago, IL, USA, 6 August 2002; Volume 1, pp. 364–368.
23. Pehlivanoglu, Y.V. A new vibrational genetic algorithm enhanced with a Voronoi diagram for path planning of autonomous UAV. *Aerosp. Sci. Technol.* **2012**, *16*, 47–55. [CrossRef]
24. Lin, Y.; Saripalli, S. Path planning using 3D dubins curve for unmanned aerial vehicles. In Proceedings of the 2014 International Conference on Unmanned Aircraft Systems (ICUAS), Orlando, FL, USA, 27–30 May 2014; pp. 296–304.
25. Yang, K.; Sukkarieh, S. Real-time continuous curvature path planning of UAVs in cluttered environments. In Proceedings of the 2008 5th International Symposium on Mechatronics and Its Applications, Amman, Jordan, 7–29 May 2008; pp. 1–6.
26. Véras, L.G.; Medeiros, F.L.; Guimarães, L.N. Rapidly exploring Random Tree* with a sampling method based on Sukharev grids and convex vertices of safety hulls of obstacles. *Int. J. Adv. Robot. Syst.* **2019**, *16*, 1729881419825941. [CrossRef]
27. Liu, Y.H.; Arimoto, S. Proposal of tangent graph and extended tangent graph for path planning of mobile robots. In Proceedings of the 1991 IEEE International Conference on Robotics and Automation, Sacramento, CA, USA, 9–11 April 1991; pp. 312–317.
28. Cover, H.; Choudhury, S.; Scherer, S.; Singh, S. Sparse tangential network (SPARTAN): Motion planning for micro aerial vehicles. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 2820–2825.
29. Babel, L. Curvature-constrained traveling salesman tours for aerial surveillance in scenarios with obstacles. *Eur. J. Oper. Res.* **2017**, *262*, 335–346. [CrossRef]
30. Yan, F.; Liu, Y.S.; Xiao, J.Z. Path planning in complex 3D environments using a probabilistic roadmap method. *Int. J. Autom. Comput.* **2013**, *10*, 525–533. [CrossRef]
31. Musliman, I.A.; Rahman, A.A.; Coors, V. Implementing 3D network analysis in 3D GIS. *Int. Arch. ISPRS* **2008**, *37 Pt B* .
32. Hart, P.E.; Nilsson, N.J.; Raphael, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [CrossRef]
33. De Filippis, L.; Guglieri, G.; Quagliotti, F. Path planning strategies for UAVS in 3D environments. *J. Intell. Robot. Syst.* **2012**, *65*, 247–264. [CrossRef]
34. Carsten, J.; Ferguson, D.; Stentz, A. 3d field d: Improved path planning and replanning in three dimensions. In Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, 9–15 October 2006; pp. 3381–3386.
35. Nash, A.; Daniel, K.; Koenig, S.; Felner, A. Theta*: Any-angle path planning on grids. In Proceedings of the AAAI 2007, Vancouver, BC, Canada, 22–26 July 2007 ; Volume 7, pp. 1177–1183.
36. Rabin, S. A* speed optimizations. *Game Program. Gems* **2000**, *1*, 272–287.
37. Ferguson, D.; Stentz, A. Field D*: An interpolation-based path planner and replanner. In *Robotics Research: Results of the 12th International Symposium ISRR*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 239–253.
38. Kim, H.; Kim, D.; Shin, J.U.; Kim, H.; Myung, H. Angular rate-constrained path planning algorithm for unmanned surface vehicles. *Ocean Eng.* **2014**, *84*, 37–44. [CrossRef]
39. Faria, M.; Marín, R.; Popović, M.; Maza, I.; Viguria, A. Efficient lazy theta* path planning over a sparse grid to explore large 3d volumes with a multirotor uav. *Sensors* **2019**, *19*, 174. [CrossRef] [PubMed]
40. Dolgov, D.; Thrun, S.; Montemerlo, M.; Diebel, J. Path planning for autonomous vehicles in unknown semi-structured environments. *Int. J. Robot. Res.* **2010**, *29*, 485–501. [CrossRef]

41. Ferraro, A.; Scordamaglia, V. A set-based approach for detecting faults of a remotely controlled robotic vehicle during a trajectory tracking maneuver. *Control Eng. Pract.* **2023**, *139*, 105655. [CrossRef]
42. Scordamaglia, V.; Nardi, V.A. A set-based trajectory planning algorithm for a network controlled skid-steered tracked mobile robot subject to skid and slip phenomena. *J. Intell. Robot. Syst.* **2021**, *101*, 15. [CrossRef]
43. Bartolomei, L.; Teixeira, L.; Chli, M. Perception-aware path planning for uavs using semantic segmentation. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020–24 January 2021; pp. 5808–5815.
44. Zhao, Y.; Yan, L.; Chen, Y.; Dai, J.; Liu, Y. Robust and efficient trajectory replanning based on guiding path for quadrotor fast autonomous flight. *Remote Sens.* **2021**, *13*, 972. [CrossRef]
45. Webb, D.J.; Berg, J.V.D. Kinodynamic RRT*: Optimal motion planning for systems with linear differential constraints. *arXiv* **2012**, arXiv:1205.5088.
46. Cain, B.; Kalaitzakis, M.; Vitzilaios, N. MK-RRT: Multi-Robot Kinodynamic RRT Trajectory Planning. In Proceedings of the 2021 International Conference on Unmanned Aircraft Systems (ICUAS), Athens, Greece, 15–18 June 2021; pp. 868–876.
47. Ge, J.; Liu, L.; Dong, X.; Tian, W. Trajectory planning of fixed-wing UAV using kinodynamic RRT algorithm. In Proceedings of the 2020 10th International Conference on Information Science and Technology (ICIST), Bath, London, Plymouth, UK, 9–15 September 2020; pp. 44–49.
48. Eun, Y.; Bang, H. Cooperative control of multiple unmanned aerial vehicles using the potential field theory. *J. Aircr.* **2006**, *43*, 1805–1814. [CrossRef]
49. Chen, X.; Zhang, J. The three-dimension path planning of UAV based on improved artificial potential field in dynamic environment. In Proceedings of the 2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics, Hangzhou, China, 26–27 August 2013; Volume 2, pp. 144–147.
50. Kitamura, Y.; Tanaka, T.; Kishino, F.; Yachida, M. 3-D path planning in a dynamic environment using an octree and an artificial potential field. In Proceedings of the Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots, Pittsburgh, PA, USA, 5–9 August 1995; Volume 2, pp. 474–481.
51. Roberge, V.; Tarbouchi, M.; Labonté, G. Fast Genetic Algorithm Path Planner for Fixed-Wing Military UAV Using GPU. *IEEE Trans. Aerosp. Electron. Syst.* **2018**, *54*, 2105–2117. [CrossRef]
52. Chai, R.; Tsourdos, A.; Savvaris, A.; Chai, S.; Xia, Y. Solving Constrained Trajectory Planning Problems Using Biased Particle Swarm Optimization. *IEEE Trans. Aerosp. Electron. Syst.* **2021**, *57*, 1685–1701. [CrossRef]
53. Dever, C.; Mettler, B.; Feron, E.; Popovic, J.; McConley, M. Nonlinear trajectory generation for autonomous vehicles via parameterized maneuver classes. *J. Guid. Control. Dyn.* **2006**, *29*, 289–302. [CrossRef]
54. Frazzoli, E.; Dahleh, M.A.; Feron, E. Real-time motion planning for agile autonomous vehicles. In Proceedings of the American Control Conference, Arlington, VA, USA, 25–27 June 2001; Volume 1, pp. 43–49.
55. Blasi, L.; Barbato, S.; D'Amato, E. A mixed probabilistic–geometric strategy for UAV optimum flight path identification based on bit-coded basic manoeuvres. *Aerosp. Sci. Technol.* **2017**, *71*, 1–11. [CrossRef]
56. Blasi, L.; D'Amato, E.; Mattei, M.; Notaro, I. UAV Path Planning in 3D Constrained Environments Based on Layered Essential Visibility Graphs. *IEEE Trans. Aerosp. Electron. Syst.* **2023**, *59*, 2359–2375. [CrossRef]
57. Al Nuaimi, M. *Analysis and Comparison of Clothoid and Dubins Algorithms for UAV Trajectory Generation*; West Virginia University: Morgantown, WV, USA, 2014 .
58. Tuttle, T.; Wilhelm, J.P. Minimal length multi-segment clothoid return paths for vehicles with turn rate constraints. *Front. Aerosp. Eng.* **2022**, *1*, 982808. [CrossRef]
59. Roskam, J. *Airplane Flight Dynamics and Automatic Flight Controls*; DARcorporation: Lawrence, Kansas, USA, 1998 .
60. Stevens, B.L.; Lewis, F.L.; Johnson, E.N. *Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems*; John Wiley & Sons: Hoboken, NJ, USA, 2015.
61. D'Amato, E.; Notaro, I.; Mattei, M. Optimal flight paths over essential visibility graphs. In Proceedings of the 2018 International Conference on Unmanned Aircraft Systems (ICUAS), Dallas, TX, USA, 12–15 June 2018; pp. 708–714.
62. Liao, B.; Hua, Y.; Wan, F.; Zhu, S.; Zong, Y.; Qing, X. Stack-RRT*: A Random Tree Expansion Algorithm for Smooth Path Planning. *Int. J. Control. Autom. Syst.* **2023**, *21*, 993–1004. [CrossRef]