4D Trajectory Planning Based on Fast Marching Square for UAV Teams

Blanca López, Javier Muñoz, Fernando Quevedo, *Graduate Student Member, IEEE*, Concepción A. Monje, *Member, IEEE*, Santiago Garrido, *Member, IEEE*, and Luis Moreno, *Member, IEEE*

***Abstract*— Swarms composed of multiple unmanned aerial vehicles (UAVs) are emerging as a key tool in various application fields including aerial surveillance, urban search and rescue, and package delivery. In this context, trajectory planning remains one of the principal concerns regarding complex robotic applications. This paper proposes a 4D path planning algorithm based on the Fast Marching Square (FM^2) method for multi-UAV teams in high-dimensional 3D environments with multiple obstacles. The 4D-FM^2 algorithm integrates a time-dependent speed function within the Fast Marching (FM) framework. The proposed algorithm was tested in a simulated urban scenario and results demonstrate that the algorithm effectively plans safe, optimal solutions to the shortest path problem for UAVs while avoiding obstacles and other drones, even in complex situations. Additionally, the algorithm excels in providing smooth speed profiles for the vehicles. Furthermore, the study also successfully evaluated the effect of various implementation parameters on the algorithm's outcome, such as the number of concurrent missions and the velocity of the vehicles. Moreover, computational time was kept within acceptable limits, and the method demonstrated overall good performance in terms of air traffic flow. The main contribution of this work is the introduction of a novel 4D trajectory planning algorithm that implicitly incorporates UAV movement and speed and hence the spatio-temporal realization of the path during the planning phase, leveraging the light propagation phenomenon. This approach efficiently addresses the presence of other UAVs and environmental obstacles, thereby preventing collisions and avoiding unnecessary airspace blockages.**

***Index Terms*— Multi-UAV systems, autonomous vehicles, fast marching, 4D trajectory planning, 3D urban environment.**

I. Introduction

UNMANNED aerial vehicles (UAVs) or drones are recently becoming popular in the need to cover and speed up a wide range of civil and commercial applications as they can perform tasks beyond the reach of human action. This is due to their ability to access and reach complex hotspots, achieve high flight speeds and perform coordinated missions together with the rest of the team. Owning to their potential, UAVs are expected to soon become an integral part of modern aviation, dominating the low-altitude airspace and overall making it safer, more affordable and much more efficient [1].

A specific type of setting where the use of this type of vehicles is of great interest and relevance is in urban scenarios. In this context, UAVs could perform search and rescue tasks [2], [3], facility inspection missions [4], delivery of goods [5] and surveillance [6], among others. However, with multiple aerial vehicles operating in the same airspace, there is a need for a strategy to manage air traffic and ensure flight safety throughout drone operations.

One of the main concerns to be solved when developing a multi-agent system is its navigation, defined as the problem of getting a given agent from an initial point to an end point, safely avoiding collisions with obstacles in the environment and with other agents, all in an autonomous way. Trajectory planning for multi-UAV teams poses challenges due to its high dimensionality and the presence of related equality and inequality constraints. Additionally, spatio-temporal cooperation is required among the drones within the swarm. In general, this problem is decomposed into two phases [7], [8]: generation of trajectories from the starting point to the goal for each vehicle (*i.e.* global path planning) and a second stage of local collision avoidance (*i.e.* conflict management stage).

These above-mentioned safety issues have so far made it impossible - and illegal - to fly drones at low altitude in most civilian situations and urban environments, as the technologies related to the remote guidance and control of drones have not reached a sufficient level of readiness to ensure the safety of people and infrastructure. Given this premise, the European association Single European Sky ATM Research (SESAR) [9] has called for "urgent action in the dimension of the airspace, in particular in the development of the so-called U-SPACE", a new framework designed to safely integrate low-level drone operations (below 120 metres flying height) into European airspace. Furthermore, given the expected increasing demand for UAVs in a shared airspace, a major concept also introduced by SESAR is the so-called 4D trajectory planning [10]. This concept aims at exploiting the idea of knowing and ensuring that a given aerial vehicle occupies a particular position within a specific time gap. This knowledge would permit a higher level of control and refinement over the trajectory computing process as potential conflicting situations could be accounted in advance and thus efficiently avoided.

Manuscript received 28 March 2023; revised 4 July 2023 and 18 September 2023; accepted 20 November 2023. Date of publication 1 December 2023; date of current version 31 May 2024. This work was supported by the EUROPEAN COMMISSION: Innovation and Networks Executive Agency (INEA), through the European H2020 LABYRINTH Project (https://labyrinth2020.eu/), under Grant H2020-MG-2019-TwoStages-861696. The Associate Editor for this article was J. W. Choi. *(Corresponding author: Blanca López.)*

The authors are with the Department of Systems Engineering and Automatics, University Carlos III of Madrid, 28911 Leganés, Spain (e-mail: bllopezp@ing.uc3m.es).

Digital Object Identifier 10.1109/TITS.2023.3336008

© 2023 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License.
For more information, see https://creativecommons.org/licenses/by/4.0/

Consequently, this 4D path planning concept would open a new scope for treating the aforementioned two path generation and obstacle avoidance stages jointly. In the literature, various approaches have been explored to tackle the issue of 4D trajectory generation for both drone navigation and civil aviation. These mainly involve optimization-based frameworks like evolutionary or swarm algorithms [11], task or departure scheduling optimization schemes [12], and receding horizon control (RHC) methods [13]. As a novel alternative, this work introduces a 4D path planning strategy for drone navigation based on Fast Marching Square (FM^2). FM^2 leverages the wavefront propagation phenomenon to compute the shortest paths or distance maps within a grid environment. The key advantage of using FM^2 as a 4D trajectory planning algorithm is its ability to implicitly consider the UAV movement and speed profile through the light propagation behaviour. This unique characteristic allows the algorithm to consider the presence of other vehicles during the planning process only and exclusively when the planned vehicle is expected to arrive at the same location simultaneously. This approach minimizes unnecessary spatial blockages in the environment and prevents undesired detours of the drone, generating smooth and conflict-free trajectories right from the planning process.

The structure of the paper is as follows. Section II details the state of the art (SoA) of trajectory planning algorithms that consider the temporal dimension of the problem. Section III describes the 4D path planning algorithm presented in this work. Sections IV and V showcase the results obtained when applying our method to UAV trajectory planning in a simulated urban scenario and provide a qualitative comparison against another closely related SoA approach. Finally, Section VI presents the main conclusions. It is worth noting that the terms drone and UAV are used indistinctly in this paper.

II. Related Work

As it was previously mentioned, two main concepts are defined around the navigation problem for UAVs: global path planning and conflict management between vehicles. These are often faced separately in the literature, although common techniques may be used for both matters. In fact, collision avoidance manoeuvres may rely on trajectory detours based on either heading or altitude changes, which can be treated as a path re-planning problem [14]. In the first phase, obstacle-free path planning is performed (considering those obstacles that are known from the environment), whilst the second phase focuses on dealing in real-time with conflict situations that cannot be avoided or predicted beforehand. Below is an analysis of strategies found in the literature to address the challenges of global and local path planning, as well as those focused on generating collision-free 4D trajectories for UAVs.

The path planning problem, particularly in complex environments with multiple constraints, variables and objectives, is often considered NP-hard [15]. However, some classic approaches used for realistic problems consider some assumptions and heuristics to reduce the complexity to that of polynomial time problems [16]. Some of the most common strategies found for solving this matter for UAVs include potential field methods [17], [18], [19], common graph search algorithms as A* [20], [21], stochastic optimization metaheuristics as evolutionary and swarm algorithms [22], [23], [24], approaches derived from dynamic programming as Dijkstra [25], [26], probabilistic methods as Probabilistic Roadmap (PRM) [27], sampling-based path planning algorithms as Rapidly-Expanding Random Tree (RRT) [21], [28] and numerical optimization approaches as either Mixed Integer Linear Programming (MILP) [29] or Model Predictive Control (MPC) [30], [31], among others. In the literature, some of these strategies have been specifically designed and implemented for the problem of coordinated path planning for multiple UAVs. A velocity-aware A* algorithm that generates paths with acceleration vectors that converge to the predefined destinations is presented in [32], whereas MPC serves as framework for nature-inspired optimization solvers in [33] and [34]. In [35] a MILP program is developed to model the problem and achieve optimal navigation of the swarm, along with two heuristic algorithms based on Dijkstra to solve the path planning problem with faster convergence speed.

Some other strategies are more specifically related to the previously mentioned collision or obstacle avoidance stage. These approaches are encompassed by the so-called reactive planning concept [16]. This term refers to the set of path planning algorithms that mostly work on the basis of local knowledge of the environment, so the goal is to prevent last-minute collisions with previously unforeseen obstacles present within a limited distance range from the vehicles. Geometric algorithms as the velocity obstacle approach [36], [37] are emerging as the main alternative for this type of planning, which relies on either Automatic Dependent Surveillance Broadcast (ADS-B) or vision-based sensing. Another approach is based on vehicle speed regulations [38]. Other options consist of those algorithms that try to reduce the needed computational cost by simplifying the process of collision avoidance to individual drone detection and dodging of obstacles, based on information captured by sensors mounted on the vehicles such as LiDARs, sonars, and radars [39]. Strategies built on UAV parameters such as the relative azimuth angle of the vehicle to the goal [40] are included in this category, whereas image processing techniques along with deep learning methods [41], [42] are a commonly used alternative.

Taking these detour decisions for collision avoidance maneuvers in real-time could result in unsolvable conflicts [43], [44]. The scalability of the UAV team and the ability to maneuver in cluttered or complex scenarios are particularly compromised by this issue [45], [46]. As conflict situations between vehicles cannot be anticipated during the initial 3D path planning phase due to their dependence on the dynamic execution of missions, efforts have been made to implicitly incorporate the time dimension into the path planning problem. These approaches, known as 4D trajectory planning algorithms, aim to enhance knowledge and control over the trajectory followed by the drones. This enables proactive conflict prevention and the generation of collision-free trajectories from the outset, resulting in improved efficiency.

Planning optimal collision-free 4D trajectories for multiple UAVs leads to optimization problems which are often faced through stochastic optimization metaheuristics, since this eliminates the local minimum problem of local optimization methods, as well as they can be run in parallel under the idea of sub-populations. Some common approaches consist on evolutionary techniques or swarm algorithms as the widely used Particle Swarm Optimization (PSO) method. In [47], the authors implement a PSO-based strategy to obtain collision-free 4D trajectories by adding one intermediate waypoint in the already computed trajectory of each UAV and changing the vehicles speed to meet the Estimated Time of Arrival (ETA) whenever a potential conflict is detected between drones. [48] follows a similar strategy for manned aircraft deconfliction also relying on adding a series of discrete waypoints to control the shape of the track and assigning arrival times to each waypoint to meet the time constraints. A Spatial Refined Voting Mechanism (SRVM) is designed for standard PSO in [49] to avoid local optimal and slow convergence and resolve the coordinated path planning problem for multiple UAVs in four-dimensional space, generating feasible flying and collision-free trajectories for each UAV from different starting points to the same destination at the same time. A similar approach can be found in [50].

Furthermore, the literature presents other ways in which this time variable is considered. One common strategy is based on treating this 4D trajectory planning as a task scheduling problem. In [51], multiple 3D paths for a drone are generated by a Clustering Improved Ant Colony Optimization (CIACO) algorithm, while at task scheduling level these results of multi-path planning for drones, along with the scheduled time information of flights and the task priorities, are regarded as the inputs for computing a 4D flight scheme by a genetic algorithm-based method. An improved 4D Ant Colony Optimization (ACO) algorithm along with a velocity optimization method can also be found in [52], which performs spatio-temporal path planning in a dynamic environment. In [53] and [54], the A* algorithm is implemented in order to solve the 4D path planning problem for a single UAV in complex dynamic scenarios. [55] presents a framework for the continuous local motion planning problem, where the receding horizon trajectories were described by 6th order Bezier polynomials and optimised via a steepest descent algorithm. Tau guidance [56] and vector-field guidance strategies [57], [58] are also found as alternative solutions to 4D trajectory generation.

In contrast to well-established control strategies and optimization-based approaches, Dougui et al. in [59] and [60] present a light propagation algorithm in order to solve potential conflicts between 4D trajectories of manned aircrafts and to avoid congested and bad-weather areas. During flight, whenever two vehicles get too close and their flight plans interfere, the resolution trajectories to solve the conflict are provided by a branch and bound (B&B) algorithm. The uncertainty related to the aircrafts position and arrival times increases the difficulty of the problem and reduces the solution space, so that the algorithm can remove 88% of the conflicts. The remaining conflicts are solved by imposing time constraints called required time of arrival. These studies are based on the wave propagation theory proposed by the Dutch scientist Christiaan Huygens [61].

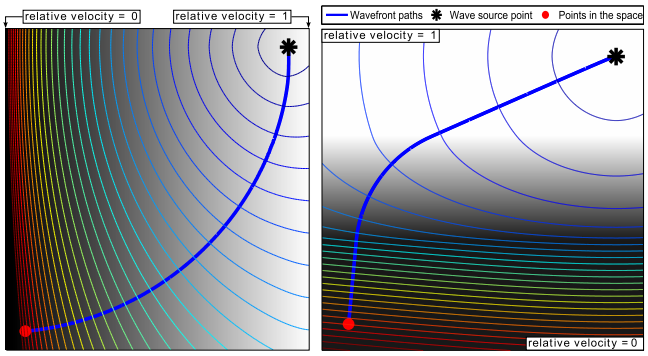


Fig. 1. Examples of a wave propagating through media with different velocities. Taken from [63].

The strategy proposed in this paper also tackles the 4D trajectory planning through a nature inspired path planning algorithm based on the light propagation phenomenon, named Fast Marching Square (FM^2). The contribution of this work with respect to these just mentioned studies is the development and implementation of a novel 4D algorithm based on a different light propagation approach, as is the subject Eikonal equation on which FM^2 relies. Preliminary research [62] suggests that algorithms using the Huygens principle are slower and less efficient than Fast Marching. Furthermore, the problem of trajectory planning will be addressed for a different type of vehicle such as UAVs and therefore we aim for application environments of distinct scale and nature like urban scenarios. In these scenarios, in addition to the presence of other team vehicles, there is the added complexity of navigating around numerous obstacles, such as buildings. As a result, the computed trajectories require a higher level of resolution compared to those needed in civil aviation. The proposed approach benefits from the inherent characteristics of the Fast Marching algorithm. It provides exact and optimal solutions to the shortest path problem in grid-based environments, ensuring correctness with respect to the chosen distance metric (*e.g.*, Euclidean distance). Additionally, the FM method exhibits deterministic behaviour, enhancing reproducibility and predictability. Its wavefront propagation and grid-based nature guarantee smooth paths, eliminating the need for extra path post-processing.

In general and among the revised literature, optimization approaches strive for global optimality in multi-vehicle path planning, allowing for custom objectives. However, they can be computationally complex. RHC algorithms, on the other hand, compute paths in real-time and excel at handling constraints. Yet, they tend to provide local optimality and require precise models. Path planning algorithms are known for computational efficiency, scalability and simplicity but may lack multi-vehicle coordination and struggle with local minima. It is important to note that the presented work addresses these two limitations. Hereinafter, the fundamentals and performance of the algorithm are introduced and analysed.

III. Methodology

In this section, we present the methods used to develop the 4D trajectory planning algorithm based on Fast Marching Square (4D-FM^2) and its application for UAV teams.

A. Motion Planning as a Light Propagation Phenomenon

A desirable feature when generating feasible trajectories for a vehicle from a starting location to a target position is to obtain smooth, short and obstacle-free paths. In nature, there are phenomena that are intended to behave in the same manner, such as light propagation. According to Fermat's principle, a light wave travelling from a point source through a material medium will reach any point in space in the fastest possible way. Thinking about it the other way around, one could reach this source along the shortest path by tracing the waves back to the goal. This behaviour becomes especially interesting for motion planning, since this artificial potential created by the wave propagation and its associated vector field gradient have the good properties desired for trajectory generation, such as smoothness and the absence of local minima.

Considering light as a linear ray and in the case of a homogeneous medium, in which the light velocity is constant (*i.e.*, constant refraction index), the light follows a straight line. When the refractive index varies continuously, the light ray is also continuously bent to avoid areas with lower light velocity (higher refractive index), whereas smooth light paths are still obtained. Examples of a light wave propagating through media with different velocities are shown in Figure 1. On this basis and from the path planning perspective, desirable characteristics for trajectory generation could be obtained by modifying the allowed light propagation speeds. For instance, no-crossing areas or obstacles of the environment could be modelled as repulsive fields by assigning a high refraction index to them and their surroundings.

B. Fast Marching and Fast Marching Square as Path Planning Methods

The Fast Marching (FM) algorithm was introduced by Sethian [64] in 1996. It is a method used to numerically solve the Eikonal equation originally on a rectangular orthogonal mesh, which models the propagation of a wave (*e.g.*, a ray of light) in a non-homogeneous medium by the arrival time of that wave at any point in space. This is given by Equation 1,

$$1 = F(\rho)|\nabla T(\rho)|, \rho \in \mathbb{R}^N, \tag{1}$$

where ρ represents any point in the environment, $T(\rho)$ is the time it would take for the wave to arrive from the initial point to point ρ, $F(\rho)$ is the velocity of the wave propagation at ρ and N constitutes the dimension of the problem. Therefore the magnitude of the gradient of the arrival function $T(\rho)$ is inversely proportional to the propagation speed. A set of multiple points ρ_0 can be determined as wave sources. As can be deduced, at any source point one has that $T(\rho_0) = 0$. This formulation is valid for multiple dimensions, since the gradient is orthogonal to the isochronal level sets of the arrival function $T(\rho)$. No reflections or interferences are considered, so the speed function never changes sign and the propagating wave only moves forward (or backwards).

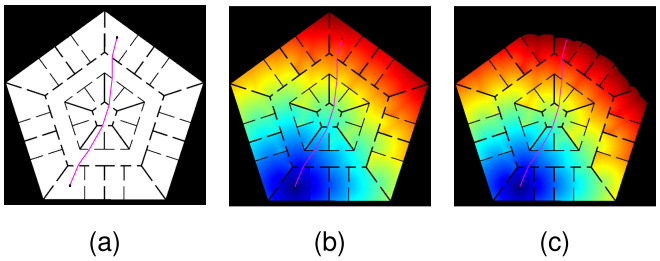


Fig. 2. Stages of the FM method. (**a**) Binary map representing $F(\rho)$ and occupancy map. (**b**) Arrival times $T(\rho)$ represented by a color map. (**c**) Wavefront propagation can be stopped when reaching goal point. Final trajectory is represented on every map.

The complete description of this algorithm along with details about its lower-level implementation is already well-explained in the literature [63]. For this reason, a broad, high-level outline of the method and its direct application to the path planning problem will be addressed in this paper. In few words, to generate a path from a starting point to a target location in either a two-dimensional or three-dimensional environment, the FM method begins by taking as input the binary occupancy matrix of the scenario. Then, it computes the arrival times from the starting point to all the points of the reachable space. Lastly, from the specified goal point and by means of gradient descent techniques, it is possible to compute the path that leads to the origin point by the shortest path.

An implementation example is shown in Figure 2 for a simulated 2D indoor scenario. In the original Fast Marching method, the wave propagation velocity $F(\rho)$ is simply discretized to zero values inside the environment obstacles and one (total velocity) in free space (see Figure 2a). Some drawbacks may derive from this. As the path obtained is the optimal in time and distance, it tends to get too close to obstacles, which would be dangerous for a vehicle to follow. Similarly, the path may not be sufficiently smooth and therefore achievable due to the vehicle kinematic constraints.

These two problems are solved in the variant Fast Marching Square (FM^2), proposed by our group [65]. As it was previously mentioned, desirable characteristics for the generated trajectories could be achieved by conveniently modifying the propagation speed values $F(\rho)$ (*i.e.*, the refraction indices) at certain locations of the space mesh. On this basis, FM^2 computes these values and makes them depend, for each point, on its distance to the nearest obstacle: grid points farther away from obstacles will allow the propagating wave to have higher velocities than points closer to obstacles. Hence, the generated paths will tend to be around areas with higher velocity values (*i.e.*, farther away from obstacles) and thus safer ones.

The FM^2 implementation consists of applying the original FM twice. Firstly, it is applied from each point in the environment that constitutes an obstacle. As a result, each point in space acquires a time value $T(\rho)$ which symbolizes the time required to reach that cell ρ from the nearest obstacle. Lower time values will mean greater proximity to obstacles. These

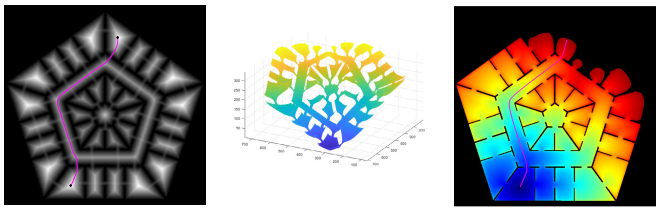


Fig. 3. Example of FM^2 implementation. (**a**) Velocity map obtained in the first FM^2 stage. (**b**) Lyapunov surface of the arrival times $T(\rho)$ in the second FM^2 stage. (**c**) Arrival times $T(\rho)$ displayed in the occupancy map. The final trajectory is represented in magenta.

values are rescaled to the interval [0,1]; therefore, the resulting grid can be represented as a grayscale map (see Figure 3a). This map can be seen as a velocity or viscosity field F: higher values of $T(\rho)$ indicate greater distances to obstacles; thus, at such points, the vehicle will be able to acquire higher velocities. Thus in the second stage, these values of $T(\rho)$ obtained in the previous stage act as propagation velocity values $F(\rho)$. Similar to the original FM method, the path is finally obtained by gradient descent techniques. An example of the FM^2 usage is shown in Figure 3, displaying a smoother and far-from-obstacles path, compared to the original FM method. In practice, the first step of FM^2 can be replaced by any function that calculates the Euclidean distance transformation of the original binary occupancy map.

Further modifications can be applied to the velocity map obtained in the first stage to obtain desired behaviours. For instance, this map can be saturated to consider higher velocities at shorter distances to obstacles. In the case of path planning for UAVs, flight levels, no-fly zones and free-flying corridors were established in some of our previous works [38], [66]. The main contributions of applying either FM or FM^2 to path planning are derived from its close relation to the light propagation phenomenon and they are summarized herein:

- **Completeness**. As the method is based on the propagation of a wave through any point in space, if there exists a path from the initial position to the goal, the method is capable of finding it.
- **No local minima**. A local minimum would imply that a point is assigned a lower $T(\rho)$ than neighbouring points closer to the wave source. This is not possible since our algorithm guarantees $F(\rho) \geq 0 \forall \rho$ and it is clearly shown in Figure 3b, where only a global minimum is observed.
- **Optimal trajectories**. The obtained path between two points in space is the optimal in both time and distance. In the case of FM^2, a smooth and safe path is obtained, which remains sufficiently optimal.
- **Fast response**. Fast Marching solvers typically have a complexity of $O(nlog(n))$, where n represents the total number of points in the considered mesh. Less common implementations may achieve linear $O(n)$ complexity [67]. This makes Fast Marching generally computationally faster than planning algorithms with exponential complexity, such as the A* algorithm, when solving certain types of problems. In terms of computational efficiency, Fast Marching would typically outperform sample methods, like the RRT algorithm, for the same high level of path optimality and quality.

Moreover, the FM method offers notable advantages when compared to potential-based algorithms such as Artificial Potential Field (APF). While both aim at yielding smooth trajectory profiles, one key advantage is its resilience to local minima. Unlike APF, which can become trapped in local minima, the FM algorithm guarantees finding the global optimum when determining the optimal path. Furthermore, the FM method inherently provides not only collision-free trajectories but also precise ETA at each waypoint. In contrast, calculating arrival times for waypoints in APF-based approaches is not as straightforward and often requires additional computational effort. Our group has extensive experience in the use of the Fast Marching method both in 2D and 3D path planning problems [68], [69], [70]. In addition, other researchers have also applied this method to their own application needs [71], [72], [73], [74].

C. 4D Fast Marching Square Trajectory Planning

The 4D trajectory planning problem can be addressed by the Fast Marching method by considering a time dependent speed function $F(\rho)$. The subsequent Eikonal equation can be written as in Equation 2,

$$1 = F(\rho, T(\rho))|\nabla T(\rho)|, \quad \rho \in \mathbb{R}^N. \tag{2}$$

Considering an increasing set of K times $t_0 < t_1 < \ldots < t_k < \ldots < t_K$ and a set of K speed maps $F_0(\rho), F_1(\rho), \ldots, F_k(\rho), \ldots, F_K(\rho)$ associated to those time steps, the Fast Marching Square algorithm can be configured to perform linear interpolation in each time interval, so that the resulting viscosity map used for trajectory planning is obtained iteratively and results in a combination of all the considered speed maps. This is expressed by Equation 3,

$$\begin{aligned} F(\rho, t) = (1-\alpha)F_k(\rho) + \alpha F_{k+1}(\rho), \\ \text{for} \quad t = (1-\alpha)t_k + \alpha t_{k+1}, \end{aligned} \tag{3}$$

where $\alpha \in [0, 1]$ and $0 \leq k < K$. For time intervals prior to t_0 and beyond t_K, $F(\rho, t) = F_0$ and $F(\rho, t) = F_K$, respectively. According to these previous statements, in order to use this 4D Fast Marching Square (4D-FM^2) approach for solving the path planning problem for multi-UAV teams, the first step consists of defining the set of times and speed maps needed.

For this application, consider the situation where a trajectory has to be planned for a UAV to perform a new mission from a start to an end point in a 3D environment. In this case, the path planner algorithm is intended to compute a conflict-free trajectory with respect to other vehicles in the swarm that will be flying over the same environment simultaneously. To accomplish this, it is essential to have knowledge of the other UAVs positions and the expected timing of their locations. For this matter, we consider the following assumptions:

- The drones considered in this study are UAVs with multiple rotors, which can climb and descend vertically.
- The drones will tend to fly at a constant velocity v_m, which corresponds to the speed assigned for the mission.

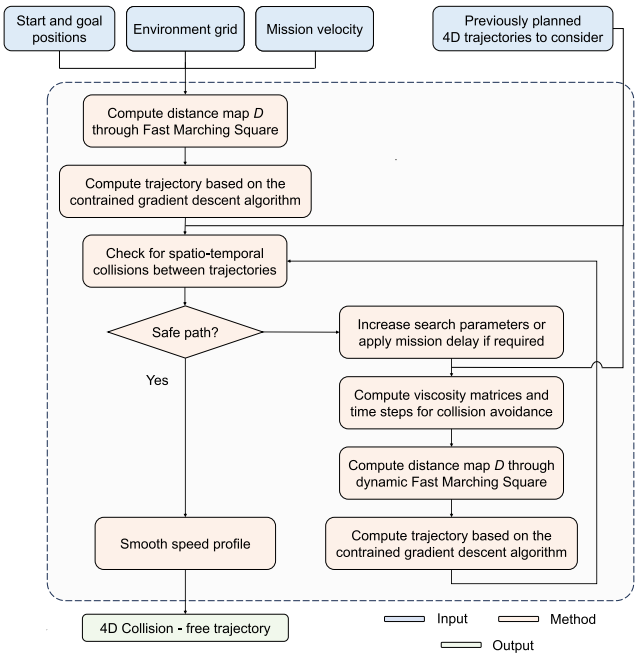


Fig. 4. Framework of the proposed 4D Fast Marching Square algorithm.

However, this cruising speed will also depend on the distance of vehicles to obstacles in the environment, modelled by $F(\rho)$ and referred to as W velocity matrix. The velocity of the UAVs may be also modified and reduced in order to avoid collisions with other drones.

- The drones are treated as a free point with three degrees of freedom. However, a safety distance is set to ensure a certain separation from buildings and other drones.

On this premise, one advantage of the FM^2 method is that it not only generates a smooth three-dimensional path, but can also provide a speed profile for the vehicle that will follow the trajectory. These velocity values can be related to our benefit to the values of the FM^2 wave propagation velocity, which can be tuned according to the viscosity map $F(\rho, t)$ (scaled from zero to one) and the desired mission speed. This way the velocity reaches its highest values (close or equal to mission speed) in the areas furthest away from the obstacles (lighter areas in Figure 3a) and other drones and minimum values in the nearest areas (darker zones in Figure 3a). Therefore, the expected time of arrival (ETA) at each waypoint (WP) of a mission trajectory can be drawn from the very own wave propagation of the method and corresponds to $T(\rho)$ of the FM^2 second stage, denoted as arrival time D matrix. The general framework of the 4D Fast Marching Square algorithm presented in this work is shown in Figure 4 and details are provided below.

1) Spatio - Temporal Coordination Between Trajectories: Going back to the scenario in which a new trajectory is to be planned for a vehicle and a set of M previously planned trajectories $E = [P_1, P_2, \ldots, P_m, \ldots, P_M]$ must be considered. From this four-dimensional data, the aforementioned associated sets of K times and speed maps are computed so that this information is considered. For instance, if another vehicle of the team is due to be flying P_{m_j} of its trajectory at time t_k, where $1 \leq j \leq NW$ and NW the number of waypoints of P_m, this three-dimensional position is marked as an obstacle (zero relative velocity value) at F_k viscosity matrix. To enhance safety, some uncertainty can be added by considering this restriction at given wider time gaps and properly registering this in the corresponding time and speed map sets. This time frame might be given by $t_k \pm t_f$, where t_f is a constant positive time value. These collections of time values and velocity matrices will henceforth be referred to as relevant time t_{set} and either viscosity or velocity F_{set} sets.

Note that the presence of another vehicle at time t_k and at a certain position of the environment will be taken into account along the planning procedure only if the vehicle to be planned is due to arrive at this location at the same time, *i.e.* if the wave propagation of the FM^2 method reaches this location accordingly. This is a significant contribution of using FM^2 as a 4D trajectory planning algorithm: the movement and speed of the vehicle and hence the spatio-temporal realization of the path can be inherently accounted in the planning phase based on the light propagation phenomenon. This way unnecessary spatial blockages of the environment can be omitted and undesired detours of the drone avoided, which would be the case if the environment space already occupied by other trajectories were blocked for new missions for any time instant.

According to this procedure, UAV trajectories are planned sequentially without any specific criteria, beyond the order of arrival of the request to calculate a new mission path. In the case no previously planned trajectories have to be contemplated, the trajectory is planned according to the basic viscosity W matrix, which considers just the distance to obstacles of the environment.

To get a robust solution, a common time reference scale can be established for all mission trajectories so that a consistent set of relevant times t_{set} is obtained. Hence, this set will be composed of time values proportional to a predefined time scale t_s. This parameter in turn enables to reduce the number of waypoints that will be eventually registered at these sets, streamlining the process of computing a new trajectory. In fact, t_s allows us to define which time and distance steps will be used to go through every mission trajectory that must be considered when generating t_{set} and F_{set} sets.

In order to guarantee a safety distance d_{sec} between drones, a security area must be conveniently marked around every waypoint position recorded in F_{set}. As it has been mentioned, other vehicles trajectory waypoints are only accurately selected and registered every t_s step, so this safety area size must be wide enough to assure d_{sec} at any time. This safety separation size d_{plus} is then given by Equation 4,

$$d_{plus} = d_{sec} + v_m \cdot t_s. \tag{4}$$

As mentioned earlier, the 4D behaviour of the Fast Marching Square algorithm presented in this work is achieved by interpolating the so-called relevant velocity matrices. In order to ensure a smooth transition, some good practice consists of *darkening* these F_{set} matrices, *i.e.* reduce its original viscosity value by a factor $0 < \varepsilon < 1$, around the potentially conflicting zones already marked with zero relative velocity value. This should be done both spatially (at neighbouring grid cells closer than a given distance d_u) and temporally wise (for $t_k \pm t_u$

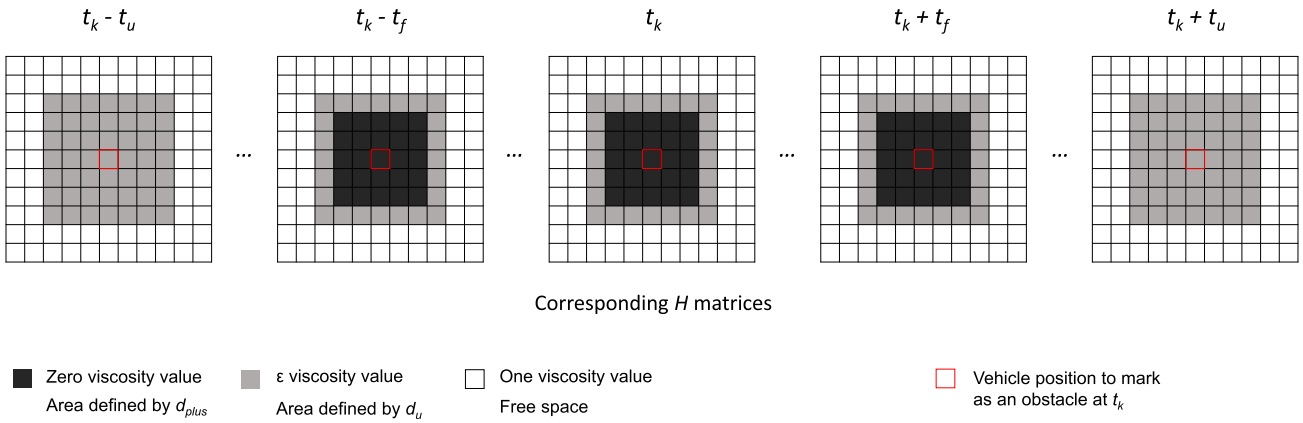


Fig. 5. 2D Representation of how the position of other vehicles is added to the path planning viscosity matrices. In this example, only one drone has to be considered at time instant t_k.

time stamps, where t_u is a constant positive time value greater than t_f). Note that time frames t_u and t_f should be set or automatically rounded up to a value proportional to t_s and that t_f must be higher than the time it takes for the vehicles to fly from a grid cell to an neighbouring one.

So far, the alleged relevant path planning sets F_{set} and t_{set} would only gather matrices informing about the presence of other vehicles and at which time instants. Hence, another effective practice involves adding extra F_k arrays equal to the basic speed map W (and their associated t_k) to the sets between every two consecutive relevant time stamps, *e.g.* t_k and t_{k+1}, that are separated for a longer time than a given threshold t_l and also at the last time stamp t_K, as no other vehicle has to be considered afterwards. This avoids unnecessary blocking of airspace during the interpolation process among F_{set} matrices. Then, the velocity matrix corresponding to t_k time instant can be defined as in Equation 5,

$$F_k = \begin{cases} v_m \cdot W \cdot H_k, & \text{if } presence \text{ at } t_k \\ v_m \cdot W, & \text{else,} \end{cases} \tag{5}$$

where $presence$ indicates the existence of other drones at time t_k and H_k is the three-dimensional matrix filled with ones in which the positions of these other vehicles (and subsequent safety areas surrounding them) are conveniently marked with either zero or ε values as appropriate. A 2D visual example of this procedure is shown in Figure 5.

2) 4D Trajectory Generation and Processing: Regarding the two stages that make up the FM^2 method, the first one would be completed once the relevant time t_{set} and velocity F_{set} sets are obtained. Then, the second phase is performed according to these sets and Equation 2. From this dynamic FM^2 implementation, the arrival times from the starting position to every point in space (D matrix) is obtained. The interpolation among velocity matrices required by this procedure has been added to our FM^2 path planner algorithm on the basis of the Fast Marching solver provided by [75].

In order to compute the trajectory from this data, gradient descent is to be performed in D. With the aim of handling the discretization of the procedure given by the discrete viscosity matrices, the Fast Marching algorithm uses an efficient numerical scheme to discretize the partial differential equation that governs the propagation of the wavefront. Then, it efficiently computes from D a continuous motion curve, which is an approximation of the geodesic path from the source point to the goal. The aforementioned FM solver includes this functionality and can provide this path solution as an output. However, for this UAV trajectory planning application in which not only spatial information on trajectories but also a profile of feasible speeds and times for the vehicles is desired, this resulting trajectory has not proved to be suitable. For the purpose of deriving these spatio-temporal trajectories, these time values have to be extracted from D by accessing the values of the matrix corresponding to the indices given by the spatial coordinates of the path. For this reason, looking at a discrete matrix through the indices of a continuous approximate geodesic path leads to velocity and time inconsistencies.

So as to solve this problem, in this work we have carried out a custom discrete trajectory search method through a constrained gradient descent algorithm. Based on D matrix and the time of arrival assigned to the goal point, the resulting path to the source location is obtained iteratively by selecting the next waypoint from the neighbouring cells of the current position through predetermined rules, listed as follows:

Rule 1: The time of arrival allocated to the next waypoint t_{next} must in any case be less than the time t_c assigned to the current position, as presented in Equation 6,

$$t_{next} < t_c. \tag{6}$$

Rule 2: The next waypoint must not be farther than a second-level neighbourhood and a solution that satisfies all criteria is first sought among the first-order adjacent cells. These concepts are illustrated in Figure 6. As a result, the maximum total distance between consecutive waypoints d_{max} is limited to less than 3.5 cells, which is the farthest possible distance from the current position.

Rule 3: Time value t_{next} must not deviate from t_c by more than the time it would take to travel d_{max} at the slowest possible speed, as shown in Equation 7,

$$|t_{next} - t_c| \leq \frac{d_{max}}{v_m \cdot \varepsilon \cdot W_{avg}}. \tag{7}$$

This is determined by v_m, factor ε, and the average viscosity value W_{avg} obtained from W for the neighbouring cells of the

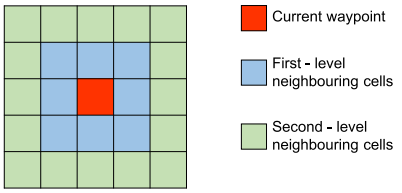


Fig. 6. 2D representation of adjacent cells to current waypoint in D matrix.

Algorithm 1 4D-FM2: Dyn. FM2 & Constrained Grad. Descent

Input: Start position, Goal position, F_{set}, t_{set}
Output: Trajectory
Compute arrival time map D from inputs via dyn. FM2
Initialize Trajectory ← Goal position and time
Current position ← Goal position and time
while Start position is not reached **do**
Current position ← neighbouring cell from current acc. to spatio-temporal Rules
Add Current position and time to Trajectory
end while
Flip Trajectory

current position (excluding those with zero value). This filters out any possible time singularities present in D.

Rule 4: Time values t_{next} and t_c must result in a reasonable travel speed v_{next} regarding an ideal velocity which considers v_m, the average viscosity value W_{next} for the current position and next waypoint cell in W and the Euclidean distance between both positions d_{next}. This is given by Equation 8,

$$v_{next} = \frac{d_{next}}{t_{next}} \approx v_i, \quad \text{where} \quad v_i = \frac{d_{next}}{v_m \cdot W_{next}}. \tag{8}$$

The first three rules serve as exclusion criteria, eliminating neighbouring cells that do not meet these requirements as potential next waypoints. The last one defines a cost function to determine the best option among the eligible adjacent cells. The cost C_q assigned to each cell, for $1 \leq q \leq Q$ where Q is the number of neighbouring cells for a current position, is determined by Equation 9,

$$C_q = |v_{iq} - v_q|, \tag{9}$$

where q stands for each neighbouring cell considered and thus, v_{iq} and v_q are the ideal and actual speed values related to each cell, respectively. The next waypoint in the path is selected as the adjacent cell q with the lowest cost. In practice, W matrix implicitly provides smooth velocity profiles, so there is no need for defining a specific criterion on that matter. This process is performed iteratively until the start point is reached and it is summarized in algorithm 1.

By following this process, a new trajectory is generated: a four-dimensional array that contains information about the spatial locations and ETA for each waypoint. From this data, an approximation of the speed v_j that the vehicle will experience at each path position in order to comply with the mission plan can be derived through Equation 10,

$$v_j = \frac{d_{j+1,j}}{t_{j+1,j}} \quad \forall\ 1 \leq j \leq NW - 1, \tag{10}$$

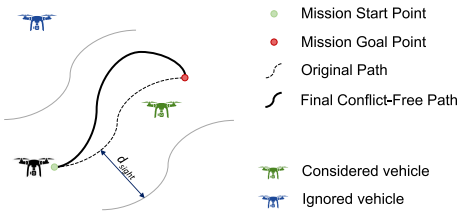


Fig. 7. Representation of mission trajectory planning when a threshold distance d_{sight} is defined to limit the number of considered vehicles.

where $d_{j+1,j}$ is the Euclidean distance between two consecutive waypoints and $t_{j+1,j}$ is the expected time that the vehicle will take to travel such distance. We consider that $v_{NW} = v_{NW-1}$. Given the arrival time map D generated by Fast Marching and the constrained gradient descent algorithm applied, the proposed method ensures the provision of the shortest possible path while guaranteeing feasible velocity dynamics for the vehicles and obstacle and collision avoidance.

As the trajectory has been extracted discretely from D, in order to ensure a smooth speed profile to avoid undesirable temporal deviations, this initial approximate profile can be slightly smoothed and the resulting arrival times can be re-calculated likewise. A fast, unsupervised and robust discretized spline smoother is used for this purpose, provided by [76].

3) Practical Considerations and Validation of Trajectories: To enhance computational speed without sacrificing path calculation quality, modifications have been made to the original approach. For instance, the wave propagation carried out in the second FM^2 stage from the initial position can be halted whenever the target location is reached, preventing the whole environment space from being assessed unnecessarily. This can be visualized in Figure 3c. In addition to this adjustment, the wave propagation may be also confined and guided through a vertical trench suitably oriented from the starting to the target point. Its width (d_w) must be enough in order to ensure that the path can be obtained within these limits.

With the aim of limiting the size of t_{set} and F_{set} and hence the amount of interpolations the method must perform to compute the mission path, a threshold distance d_{sight} can be set so as to narrow the number of other vehicle positions considered at path planning level. To do so, the whole procedure to compute a trajectory can be split into two calls to the FM solver. First, an ideal trajectory is calculated which does not consider the presence of any other vehicle. In the latter stage, only vehicles closer than d_{sight} to this base path are included in the relevant time and viscosity sets. This can be visualised in Figure 7. It may occur that in this second phase, no other trajectories need to be contemplated. In this scenario, the final trajectory will be the base path already calculated and there would be no need for a second call to the solver.

Similarly, trajectories are expected to preferably present horizontal displacements, with respect to flight altitude variations. For this reason, safety distances d_{plus} and d_u used to mark a vehicle position in F_{set} matrices can be reduced on the vertical axis by a factor λ, where $0 < \lambda < 1$. In order to ensure that a new mission plan is valid and does not interfere with other previously computed trajectories, it is expressly checked that

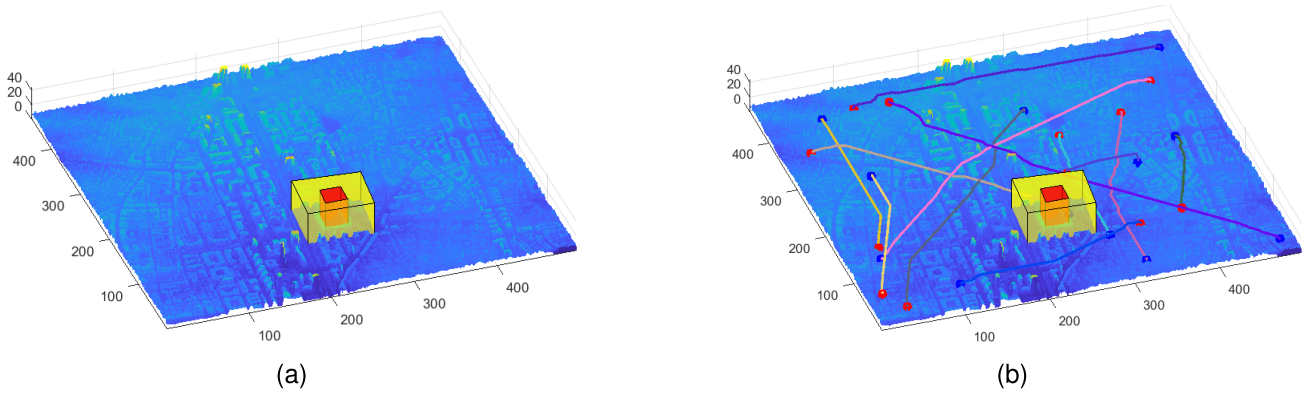


Fig. 8. Simulation environment. (**a**) Surroundings of the Santiago Bernabéu Stadium area in Madrid (Spain). Grid cell resolution is 5 meters. (**b**) Example set of 12 mission trajectories planned in the simulated scenario.

Algorithm 2 Generate Viscosity Matrices & Time Steps for CA

Input: Base FM2 Traj., W, v_m, 4D Traj. to avoid, Dep. time, Safety param.: t_f, t_s, d_{plus}, ε, t_u, d_u, d_{sight}, λ, t_l
Output: Viscosity F_{set} and time t_{set} relevant sets
Initialize H occupancy matrices
for every WP of each 4D Trajectory **do**
  if Pos. within d_{sight} from Base Traj. **&** Time $\geq$ Dep. time **&** (Prev. saved WP Time - Time) $\geq t_s$ **then**
    Mark 3D location in corresponding H matrices acc. to Safety param. and Time (see Fig. 5)
  end if
end for
Compute F_{set} & t_{set} acc. to H, v_m, W, t_l (see Eq. 5)

no waypoint in the new path is at a shorter distance than d_{sec} spatially and closer than t_f temporally wise. Working with a too restrictive d_{sight} or λ might result in a conflicting plan. These instances are detected and planning for a new trajectory is attempted for either wider d_{sight} or $\lambda = 1$ values.

When computing a path for a new mission, it is intended that its corresponding time plan commences as soon as possible regarding a given take-off time value. There may be cases where this is not feasible, *e.g.* when the area around the initial waypoints is occupied by other vehicles at that time instant. In addition, other blockages marked in the viscosity matrices F_{set} can cause that for such launch time it is not possible to find a correct trajectory as no other alternative path can be found to properly dodge these occlusions. This problem is avoided to a large extent by the aforementioned *darkening* process carried out by factor ε, but not completely. A mission temporal delay t_d is applied in these situations. Algorithm 2 outlines the pseudocode for generating F_{set} and t_{set} for spatio-temporal coordination along with some of these considerations.

IV. Results

In order to test the 4D Fast Marching Square planner presented in this work, a series of simulations are carried out to analyze and verify its performance. The designed algorithm is implemented in MATLAB and the tests are conducted on a computer with 16 GB of RAM and an AMD Ryzen 5 3600 CPU running Windows 10. The results for these simulations are presented and analysed in this section.

The experiments are performed for varying numbers of simultaneous missions: 5, 10, 15, and 20. Each one of these missions corresponds to a UAV vehicle flying from an initial position to a target location. For this reason, it is interchangeable to refer to either the number of drones or the number of simultaneous missions. For each simulation, 7200 iterations of simulation are accomplished, with each one equivalent to 1 second of actual flight time. When a mission is completed, a new one is planned for the corresponding vehicle. Each set of simultaneous missions is studied for performance at different mission speeds v_m, including 2, 4, and 6 cells/s. Therefore, there are a total of 12 distinct executions to fully test the algorithm. For the simulated movement of the drones, it has been assumed that they fly following the velocity profile previously computed based on the ETA obtained during the trajectory generation process. It has been considered that the vehicles have a maximum acceleration of 1.25 cells/s^2 in any direction. Therefore, in each iteration of the algorithm, the vehicles can adjust their velocity to the said profile, taking into account this speed variation restriction, and accordingly advance spatially in their trajectory. Planning a new trajectory is considered a separate and premeditated process, and it is consequently not included in the 2 hours of simulated flight time. In reality, the creation of a new trajectory will take into account the positions and ETA of other drones that may affect the current trajectory, based on the expected take-off time. Hence, these data are taken into consideration as input values during the planning stage.

The initial and final points of each mission trajectory are arbitrarily generated, from any position of free space. The simulated environment for the tests is shown in Figure 8a, which corresponds to the surroundings of the Santiago Bernabéu Stadium area in Madrid (Spain). It presents a resolution of 5 meters per cell over a 467 $\times$ 497 $\times$ 44 matrix. In this urban scenario, the robustness of the algorithm against planning in large hindered environments will be tested. Specifically, the trajectory generation process will have to deal with a vast number of buildings and also consider two restricted and no-fly zones (represented as yellow and red cuboids, respectively).

This restricted area states that it should only be crossed if necessary and avoided as far as possible. An example set of 12 mission trajectories is shown in Figure 8b.

The relevant parameters discussed in the previous sections have been chosen based on experimentation. Safety distance d_{sec} is set to 7 cells; t_s (used to uniformly determine t_{set}) is defined to 2 seconds for experiments with v_m equal to 2 cell/s and 1.25 seconds otherwise; t_f (used to deal with uncertainties) is set to 5 seconds, while t_u (used for smooth interpolation of F_{set}) equals 8 seconds. The *darkening* factor ε is defined to 0.65, whereas safety distance d_u is set to $1.5 \cdot d_{sec}$. Time constant t_l (used to avoid unnecessary airspace blockage) is $5 \cdot t_s$. Distances d_w and d_{sight}, which ease computational load, are defined to 200 and 20 cells, correspondingly and according to the dimensions of the simulation environment, and mission delay time t_d is 20 seconds. If d_{sight} constitutes a too restrictive value in order to find a feasible trajectory, its value is doubled in the next try. For the simulations referred to in this section, it has been established that this incremental process may be attempted a maximum of 3 times, *i.e.* the highest value for d_{sight} is 60 cells. Similarly, a mission may be delayed by a maximum of 2 minutes. Factor λ equals 0.67. If a trajectory cannot be computed for a given mission based on these parameters and conditions, the mission shall be considered failed and a new one is assigned.

A series of statistical measures are collected to check the performance of the strategy. These measures are taken for each of the 12 aforementioned algorithm executions and are as follows: Travelled distance per trajectory, in cells; Average flying speed during mission, in cells/s; Computational time to plan a mission trajectory, in seconds; Required mission delay time to find a feasible trajectory, in seconds; Number of drones simultaneously flying per iteration; Number of missions accomplished by each UAV individually; Total number of planned missions for the whole team; Minimum separation distance between vehicles, in cells; Number of failed missions when trajectory planning; Number of planned missions requiring 4D-FM^2 second stage; and lastly, checking distance d_{sight} required for 4D-FM^2 second stage.

The data resulting from the simulation runs are shown in Figures 9–11. For each mission speed v_m and for each set of simultaneous missions, some of the aforementioned measurements are presented right away and others in the form of average value. Besides, the dispersion of the obtained results is given by standard deviation. Negative values coming from these dispersion measures should not lead to misunderstandings. The simulation data corresponding to every v_m used are represented beside one another, around the vertical axis associated to the number of drones of the team. From left to right, data belong to simulations carried out with 2, 4 and 6 cells/s, respectively. Hereafter, some comments about the obtained data are expressed.

In Figure 9a, it becomes evident that as the speed v_m assigned to every mission increases, the number of missions completed by each vehicle also increases proportionally. This is because the vehicles can complete the tasks in less time. However, this measure remains almost constant in terms of the number of drones in the swarm. It is important to note that

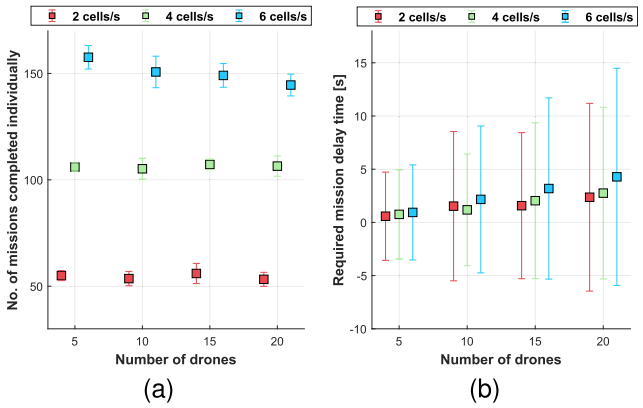


Fig. 9. Statistical measures collected through simulations. (**a**) No. of missions accomplished by each UAV. (**b**) Delay time required for a mission.

for higher speeds, the number of missions performed by each vehicle starts to trend slightly downwards. Figure 9b provides an explanation for this observation: as the number of drones in the team and the speed increases, the waiting time to start a mission gradually increases, although it is always maintained at low and reasonable values. In addition, while the number of vehicles that can fly at the same time is always close to the maximum value, it decreases inversely to v_m (see Figure 10a). Therefore, it can be concluded that the higher the speed and the number of simultaneous missions, the more congested the airspace becomes, making it increasingly complex to manage.

A similar trend is shown in Figure 10b. Checking distance d_{sight} is always kept around the default value (20 cells). However, it is observed that as traffic congestion increases, it becomes necessary to increase the value of the parameter. In addition, Figure 10c shows that the number of missions requiring the second stage of the method increases both with the number of drones in the swarm and the speed of the missions. Although these factors suggest that generating a new trajectory becomes more challenging, the total number of missions performed by the entire group increases in direct proportion to both criteria (see Figure 10d). These last two measures must be analyzed jointly.

In Figure 11a, the length of the trajectories remains constant regardless of the number of drones and the speed v_m, as the start and end points of the missions are randomly generated. Similarly, the minimum distance between a pair of UAVs depends on the randomness of the missions, but it is reasonable to expect that the minimum distance decreases as traffic congestion increases (see Figure 11b). Note that safety distance d_{sec} has been maintained throughout all of the experiments.

With regards to the time needed to compute a new trajectory, Figure 11c indicates that this measure proportionally increases to the number of drones, as more trajectories have to be considered during planning. However, this time value is lower for higher speeds. This is because the wave propagation carried out in the 4D-FM^2 method can be performed at a faster pace, speeding up the computational process. Therefore, it has been found that a new mission can generally be planned by the 4D-FM^2 algorithm in less than 25 seconds, even in the least

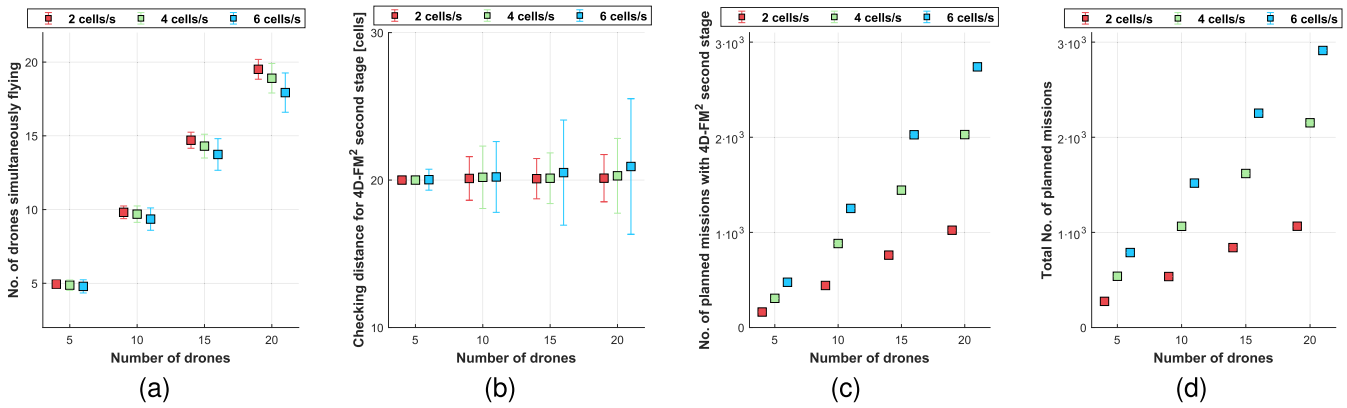


Fig. 10. Statistical measures collected through simulations. (**a**) No. of UAVs flying per iteration. (**b**) Checking distance d_{sight} required. (**c**) Number of missions needing 4D-FM^2 second stage. (**d**) Total No. of missions planned for the team.

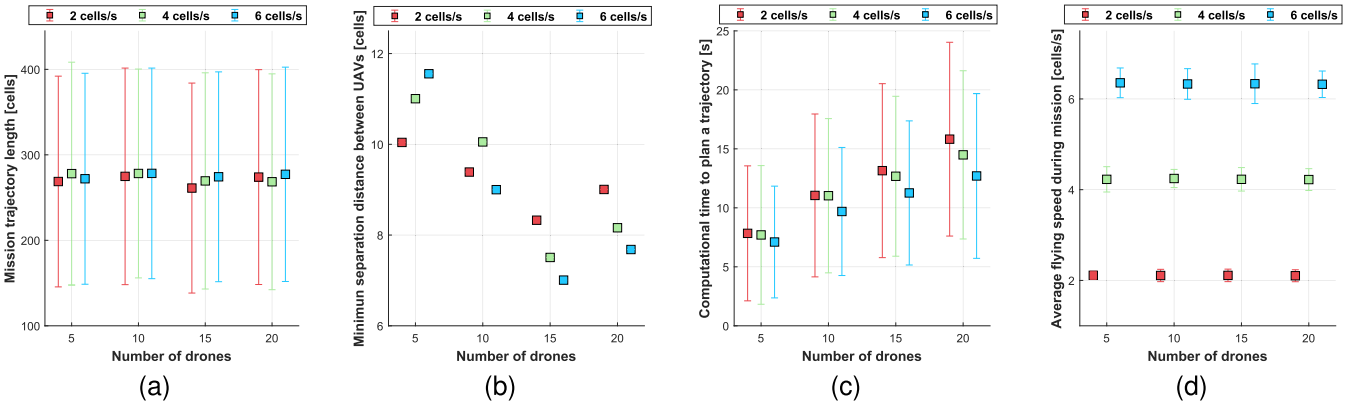


Fig. 11. Statistical measures collected through simulations. (**a**) Mission trajectory length. (**b**) Minimum separation distance given between UAVs. (**c**) Computational time to plan a mission trajectory. (**d**) Average flying speed during mission.

favorable case (*i.e.* highest number of vehicles and lowest speed), with the overall average time being less than this value.

This suggests that the original $O(nlog(n))$ time complexity of the FM^2 method, as well as the memory usage, are influenced in this 4D-FM^2 version by the use of a dynamic viscosity matrix, which enables to introduce the time dimension. The additional complexity is linear and not excessively large due to the simple linear interpolation among viscosity matrices. However, quantifying this increase is challenging as it depends on the number of drone trajectories to consider and, consequently, the number of viscosity matrices to handle. Furthermore, it is worth noting that given the practical considerations described in Section III, some trajectories can be planned without the need to explicitly take other drone paths into account. In such cases, trajectory planning can be performed using a general Fast Marching Square call, reducing the complexity associated with the viscosity matrix handling.

Finally, Figure 11d shows that the speed at which vehicles perform their missions, obtained from the ETA values given by matrix D, is indeed very close to the established velocity v_m. However, it is observed that these speed values are slightly higher than expected. This may be due to inaccuracies inherent in the Fast Marching solver used. According to documentation provided by [75], areas marked with zero viscosity values in F_{set} matrices (in our case, due to obstacles or other vehicles) may cause a source of numerical uncertainty in the wave propagation calculation of the FM method.

Given that missions were randomly generated and subject to constraints on maximum mission delay times and d_{sight}, mission planning failures could be expected. When v_m was set to 2 cells/s, all missions were computed successfully without any issues. For higher speeds, just up to 2 failed missions occurred for each set of simultaneous missions, which can be regarded as negligible considering the total number of missions effectively computed through each simulation test.

V. Further Discussion

This section covers several additional aspects of the proposed algorithm's performance. In order to provide a qualitative comparison to the aforementioned work presented in [60], we tested the 4D-FM^2 algorithm in a 2D environment used as benchmark by the cited authors. In this case, 7 aircrafts perform a conflicting mission which converges towards the same point. These are initially located on a circle of radius 100 Nm. The cited authors use an environment grid scale equivalent to the standard horizontal separation unit in aviation (5 Nm) and the vehicles velocity is 450 kts.

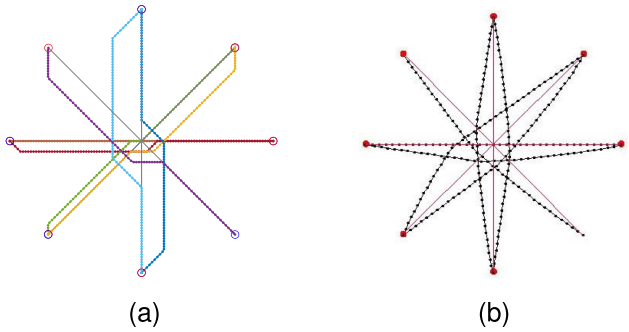


Fig. 12. Light propagation algorithms applied to 2D trajectory planning. (**a**) Our 4D-FM^2 algorithm (**b**) Method provided by [60].

To assess our proposed algorithm and regarding the earlier stated characteristic simulation parameters, for this test these have been selected according to the values provided by the comparative study. We used a grid scale of 1 Nm to provide higher resolution, and hence v_m is conveniently set to 0.125 cells/s. Safety distance d_{sec} is set to 5 cells (equal to 1 standard separation unit in aviation). Although the following parameters are not specifically determined in the benchmark survey, we chose to define t_s as 3 seconds, t_f is set to 12 seconds, which corresponds to the minimum time it will take for a vehicle to fly to a neighbouring cell according to v_m plus a threshold of 4 seconds, and t_u equals 15 seconds. Given the reduced dimensions of the environment and the distance unit used, a limiting distance d_w has not been established and d_{sight} is set to 10 cells. The rest of the parameters keep the same value as in previous tests. Figure 12 shows the resulting trajectories.

According to the selected parameters, there has been no need to delay any mission to guarantee a collision-free trajectory and all missions required 4D-FM^2 second stage except for the first one. The figure shows that the resulting paths attempt to adhere as closely as feasible to the ideal trajectory (a straight line between the starting and ending points, represented in gray color) while keeping the highest and safest distance to other vehicles. The set of 7 trajectories was computed in under 2.5 seconds in terms of computational cost. Although the various machine characteristics could not be directly compared, the computational time obtained is lower than the $\approx$ 30 seconds corresponding value reported by Dougui et al.. Hence, the Fast Marching method *a priori* achieves a more efficient solution to the path planning problem based on the light propagation phenomenon than the one based on the Huygens principle.

In addition to the simulation results discussed earlier, Figure 13 illustrates examples of velocity profiles generated by the 4D-FM^2 method for the tested set of velocities v_m and the first missions to be performed by a team of 10 drones. The method has been found to produce adequately smooth speed profiles, particularly for lower mission velocities. This was evidenced throughout the simulation runs, where vehicles were able to follow their designated trajectories without encountering any conflicts and maintain a safe distance at any time.

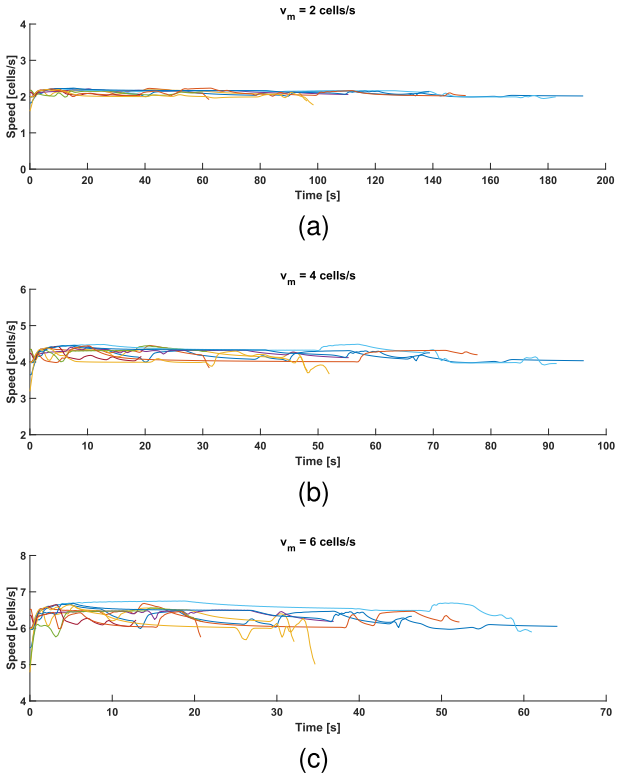


Fig. 13. Examples of speed profiles generated by 4D-FM^2 algorithm for the set of tested mission velocities v_m and a team of 10 drones: (**a**) 2 cells/s, (**b**) 4 cells/s and (**c**) 6 cells/s.

VI. Conclusion

The main objective of this work was to novelly introduce a trajectory planning algorithm based on the light propagation phenomenon described by the FM^2 method. This has been achieved by considering a time dependent speed function. This way, the presence of other vehicles can be considered at corresponding time intervals, allowing to dodge other drone locations during the planning process just when simultaneous spatio-temporal overlap is expected. This represents a significant contribution of our approach: by implicitly considering the spatio-temporal movement of the vehicles through the algorithm's wave expansion, our method allows for the omission of unnecessary spatial blockages in the environment and the avoidance of undesired detours for the drones.

The basis of the proposed method was presented and how other previously planned trajectories are handled to compute a collision-free plan is described. A constrained gradient descent algorithm based on predefined spatio-temporal rules was developed to obtain the resulting trajectories with the aim of obtaining a smooth speed profile for the vehicles to follow, while simultaneously ensuring collision avoidance and optimality in terms of both travelled distance and time.

The 4D Fast Marching Square (4D-FM^2) method was evaluated in a high-dimensional 3D simulated urban scenario with multiple obstacles for various combinations of concurrent missions and UAV velocities. The results demonstrated that the minimum safety distance was consistently

maintained throughout the simulations. Moreover, the computational time remained within acceptable limits, and the method demonstrated overall good performance in terms of air traffic flow. Additionally, the effectiveness of the algorithm was demonstrated in a 2D scenario, proving the adaptability of the method to use cases of different scale and requirements.

Some future lines of research for this work could involve the development of a non-sequential approach, in order to consider an equitable order of preference among missions that are to be planned jointly. Furthermore, other Fast Marching solvers could be tested for this 4D trajectory planning implementation in the hope of coming up with a faster solution while guaranteeing smooth speed profiles and robust performance.

REFERENCES

[1] N. S. Labib, M. R. Brust, G. Danoy, and P. Bouvry, "The rise of drones in Internet of Things: A survey on the evolution, prospects and challenges of unmanned aerial vehicles," *IEEE Access*, vol. 9, pp. 115466–115487, 2021.

[2] J. Scherer et al., "An autonomous multi-UAV system for search and rescue," in *Proc. 1st Workshop Micro Aerial Vehicle Netw., Syst., Appl. Civilian Use*, May 2015, pp. 33–38.

[3] E. T. Alotaibi, S. S. Alqefari, and A. Koubaa, "LSAR: Multi-UAV collaboration for search and rescue missions," *IEEE Access*, vol. 7, pp. 55817–55832, 2019.

[4] V. T. Hoang, M. D. Phung, T. H. Dinh, and Q. P. Ha, "System architecture for real-time surface inspection using multiple UAVs," *IEEE Syst. J.*, vol. 14, no. 2, pp. 2925–2936, Jun. 2020.

[5] R. Shakeri et al., "Design challenges of multi-UAV systems in cyber-physical applications: A comprehensive survey and future directions," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3340–3385, 4th Quart., 2019.

[6] J. Scherer and B. Rinner, "Persistent multi-UAV surveillance with energy and communication constraints," in *Proc. IEEE Int. Conf. Autom. Sci. Eng. (CASE)*, Aug. 2016, pp. 1225–1230.

[7] M. Radmanesh, M. Kumar, P. H. Guentert, and M. Sarim, "Overview of path-planning and obstacle avoidance algorithms for UAVs: A comparative study," *Unmanned Syst.*, vol. 6, no. 2, pp. 95–118, Apr. 2018.

[8] S. Huang, R. S. H. Teo, and K. K. Tan, "Collision avoidance of multi unmanned aerial vehicles: A review," *Annu. Rev. Control*, vol. 48, pp. 147–164, Jan. 2019.

[9] B. Van Houtte, "The single European sky—Eu reform of ATM," in *European Air Traffic Management*. Evanston, IL, USA: Routledge, 2016, pp. 200–217.

[10] G. Enea and M. Porretta, "A comparison of 4D-trajectory operations envisioned for Nextgen and SESAR, some preliminary findings," in *Proc. 28th Congr. Int. Council Aeronaut. Sci.*, vol. 5. Edinburgh, U.K.: Optimage Ltd., 2012, pp. 4152–4165.

[11] T. Guo, Y. Mei, K. Tang, and W. Du, "A knee-guided evolutionary algorithm for multi-objective air traffic flow management," *IEEE Trans. Evol. Comput.*, early access, Jun. 1, 2023, doi: 10.1109/TEVC.2023.3281810.

[12] Y. Chen, Y. Xu, M. Hu, F. Huang, and Q. Nie, "A 4D-trajectory planning method based on hybrid optimization strategy for demand and capacity balancing," in *Proc. IEEE/AIAA 40th Digit. Avionics Syst. Conf. (DASC)*, Oct. 2021, pp. 1–9.

[13] D. B. Seenivasan, A. Olivares, and E. Staffetti, "Multi-aircraft optimal 4D online trajectory planning in the presence of a multi-cell storm in development," *Transp. Res. C, Emerg. Technol.*, vol. 110, pp. 123–142, Jan. 2020.

[14] J. Chen, Y. Zhou, Q. Lv, K. K. Deveerasetty, and H. U. Dike, "A review of autonomous obstacle avoidance technology for multi-rotor UAVs," in *Proc. IEEE Int. Conf. Inf. Autom. (ICIA)*, Aug. 2018, pp. 244–249.

[15] J. Yu and S. M. LaValle, "Optimal multirobot path planning on graphs: Complete algorithms and effective heuristics," *IEEE Trans. Robot.*, vol. 32, no. 5, pp. 1163–1177, Oct. 2016.

[16] C. Goerzen, Z. Kong, and B. Mettler, "A survey of motion planning algorithms from the perspective of autonomous UAV guidance," *J. Intell. Robotic Syst.*, vol. 57, nos. 1–4, pp. 65–100, Jan. 2010.

[17] H. M. Jayaweera and S. Hanoun, "A dynamic artificial potential field (D-APF) UAV path planning technique for following ground moving targets," *IEEE Access*, vol. 8, pp. 192760–192776, 2020.

[18] H. Heidari and M. Saska, "Collision-free trajectory planning of multi-rotor UAVs in a wind condition based on modified potential field," *Mechanism Mach. Theory*, vol. 156, Feb. 2021, Art. no. 104140.

[19] Z. Pan, C. Zhang, Y. Xia, H. Xiong, and X. Shao, "An improved artificial potential field method for path planning and formation control of the multi-UAV systems," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 69, no. 3, pp. 1129–1133, Mar. 2022.

[20] D. Mandloi, R. Arya, and A. K. Verma, "Unmanned aerial vehicle path planning based on A* algorithm and its variants in 3D environment," *Int. J. Syst. Assurance Eng. Manage.*, vol. 12, no. 5, pp. 990–1000, Oct. 2021.

[21] C. Zammit and E.-J. van Kampen, "Comparison between A* and RRT algorithms for 3D UAV path planning," *Unmanned Syst.*, vol. 10, no. 02, pp. 129–146, Apr. 2022.

[22] J. D. S. Arantes, M. D. S. Arantes, C. F. M. Toledo, O. T. Júnior, and B. C. Williams, "Heuristic and genetic algorithm approaches for UAV path planning under critical situation," *Int. J. Artif. Intell. Tools*, vol. 26, no. 01, Feb. 2017, Art. no. 1760008.

[23] Y. V. Pehlivanoglu and P. Pehlivanoglu, "An enhanced genetic algorithm for path planning of autonomous UAV in target coverage problems," *Appl. Soft Comput.*, vol. 112, Nov. 2021, Art. no. 107796.

[24] M. D. Phung and Q. P. Ha, "Safety-enhanced UAV path planning with spherical vector-based particle swarm optimization," *Appl. Soft Comput.*, vol. 107, Aug. 2021, Art. no. 107376.

[25] L. Deng, H. Yuan, L. Huang, S. Yan, and Y. Lai, "Post-earthquake search via an autonomous UAV: Hybrid algorithm and 3D path planning," in *Proc. 14th Int. Conf. Natural Comput., Fuzzy Syst. Knowl. Discovery (ICNC-FSKD)*, Jul. 2018, pp. 1329–1334.

[26] Z. He and L. Zhao, "The comparison of four UAV path planning algorithms based on geometry search algorithm," in *Proc. 9th Int. Conf. Intell. Hum.-Mach. Syst. Cybern. (IHMSC)*, vol. 2, Aug. 2017, pp. 33–36.

[27] Z. Xu, D. Deng, and K. Shimada, "Autonomous UAV exploration of dynamic environments via incremental sampling and probabilistic roadmap," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2729–2736, Apr. 2021.

[28] J. Xiong and X. Duan, "Path planning for UAV based on improved dynamic step RRT algorithm," *J. Phys., Conf. Ser.*, vol. 1983, no. 1, Jul. 2021, Art. no. 012034.

[29] W. Stecz and K. Gromada, "UAV mission planning with SAR application," *Sensors*, vol. 20, no. 4, p. 1080, Feb. 2020.

[30] D. Yan, W. Zhang, H. Chen, and J. Shi, "Robust control strategy for multi-UAVs system using MPC combined with Kalman-consensus filter and disturbance observer," *ISA Trans.*, vol. 135, pp. 35–51, Apr. 2023.

[31] B. Lindqvist, S. S. Mansouri, A.-A. Agha-Mohammadi, and G. Nikolakopoulos, "Nonlinear MPC for collision avoidance and control of UAVs with dynamic obstacles," *IEEE Robot. Autom. Lett.*, vol. 5, no. 4, pp. 6001–6008, Oct. 2020.

[32] Y. Hu, Y. Yao, Q. Ren, and X. Zhou, "3D multi-UAV cooperative velocity-aware motion planning," *Future Gener. Comput. Syst.*, vol. 102, pp. 762–774, Jan. 2020.

[33] Y. Wu, Y. Wang, X. Qu, and L. Sun, "Exploring mission planning method for a team of carrier aircraft launching," *Chin. J. Aeronaut.*, vol. 32, no. 5, pp. 1256–1267, May 2019.

[34] P. Yao, H. Wang, and H. Ji, "Multi-UAVs tracking target in urban environment by model predictive control and improved grey wolf optimizer," *Aerosp. Sci. Technol.*, vol. 55, pp. 131–143, Aug. 2016.

[35] A. Bahabry, X. Wan, H. Ghazzai, H. Menouar, G. Vesonder, and Y. Massoud, "Low-altitude navigation for multi-rotor drones in urban areas," *IEEE Access*, vol. 7, pp. 87716–87731, 2019.

[36] A. Lombard, L. Durand, and S. Galland, "Velocity obstacle based strategy for multi-agent collision avoidance of unmanned aerial vehicles," in *Proc. IEEE Int. Conf. Sens., Commun. Netw. (SECON Workshops)*, Jun. 2020, pp. 1–6.

[37] C. Y. Tan, S. Huang, K. K. Tan, and R. S. H. Teo, "Three dimensional collision avoidance for multi unmanned aerial vehicles using velocity obstacle," *J. Intell. Robot. Syst.*, vol. 97, no. 1, pp. 227–248, Jan. 2020.

[38] B. López, J. Muñoz, F. Quevedo, C. A. Monje, S. Garrido, and L. E. Moreno, "Path planning and collision risk management strategy for multi-UAV systems in 3D environments," *Sensors*, vol. 21, no. 13, p. 4414, Jun. 2021.

[39] J. N. Yasin, S. A. S. Mohamed, M.-H. Haghbayan, J. Heikkonen, H. Tenhunen, and J. Plosila, "Unmanned aerial vehicles (UAVs): Collision avoidance systems and approaches," *IEEE Access*, vol. 8, pp. 105139–105155, 2020.

[40] S. Zhang, M. Xu, and X. Wang, "Research on obstacle avoidance algorithm of multi-UAV consistent formation based on improved dynamic window approach," in *Proc. IEEE Asia–Pacific Conf. Image Process., Electron. Comput. (IPEC)*, Apr. 2022, pp. 991–996.

[41] D. Wang, W. Li, X. Liu, N. Li, and C. Zhang, "UAV environmental perception and autonomous obstacle avoidance: A deep learning and depth camera combined solution," *Comput. Electron. Agricult.*, vol. 175, Aug. 2020, Art. no. 105523.

[42] H. Y. Lee, H. W. Ho, and Y. Zhou, "Deep learning-based monocular obstacle avoidance for unmanned aerial vehicle navigation in tree plantations: Faster region-based convolutional neural network approach," *J. Intell. Robot. Syst.*, vol. 101, no. 1, pp. 1–18, Jan. 2021.

[43] E. Ferrera, A. Alcántara, J. Capitán, A. Castaño, P. Marrón, and A. Ollero, "Decentralized 3D collision avoidance for multiple UAVs in outdoor environments," *Sensors*, vol. 18, no. 12, p. 4101, Nov. 2018.

[44] C. Luo, S. I. McClean, G. Parr, L. Teacy, and R. De Nardi, "UAV position estimation and collision avoidance using the extended Kalman filter," *IEEE Trans. Veh. Technol.*, vol. 62, no. 6, pp. 2749–2762, Jul. 2013.

[45] Z. Liu, Y. Zhang, C. Yuan, L. Ciarletta, and D. Theilliol, "Collision avoidance and path following control of unmanned aerial vehicle in hazardous environment," *J. Intell. Robot. Syst.*, vol. 95, no. 1, pp. 193–210, Jul. 2019.

[46] Y. Du, X. Zhang, and Z. Nie, "A real-time collision avoidance strategy in dynamic airspace based on dynamic artificial potential field algorithm," *IEEE Access*, vol. 7, pp. 169469–169479, 2019.

[47] D. Alejo, J. A. Cobano, G. Heredia, and A. Ollero, "Collision-free 4D trajectory planning in unmanned aerial vehicles for assembly and structure construction," *J. Intell. Robot. Syst.*, vol. 73, nos. 1–4, pp. 783–795, Jan. 2014.

[48] S. Chaimatanan, D. Delahaye, and M. Mongeau, "A hybrid Metaheuristic optimization algorithm for strategic planning of 4D aircraft trajectories at the continental scale," *IEEE Comput. Intell. Mag.*, vol. 9, no. 4, pp. 46–61, Nov. 2014.

[49] Y. Liu, X. Zhang, Y. Zhang, and X. Guan, "Collision free 4D path planning for multiple UAVs based on spatial refined voting mechanism and PSO approach," *Chin. J. Aeronaut.*, vol. 32, no. 6, pp. 1504–1519, Jun. 2019.

[50] H.-B. Duan, X.-Y. Zhang, J. Wu, and G.-J. Ma, "Max-min adaptive ant colony optimization approach to multi-UAVs coordinated trajectory replanning in dynamic and uncertain environments," *J. Bionic Eng.*, vol. 6, no. 2, pp. 161–173, Jun. 2009.

[51] Y. Wu, K. H. Low, B. Pang, and Q. Tan, "Swarm-based 4D path planning for drone operations in urban environments," *IEEE Trans. Veh. Technol.*, vol. 70, no. 8, pp. 7464–7479, Aug. 2021.

[52] H. Shi, Y. Lu, Z. Hou, and S. Huang, "Research on UAV 4D path planning and velocity optimization method," in *Proc. Int. Comput., Signals Syst. Conf. (ICOMSSC)*, Sep. 2018, pp. 876–880.

[53] P. P.-Y. Wu, D. Campbell, and T. Merz, "Multi-objective four-dimensional vehicle motion planning in large dynamic environments," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 41, no. 3, pp. 621–634, Jun. 2011.

[54] H. Cicibas, K. A. Demir, and N. Arica, "Comparison of 3D versus 4D path planning for unmanned aerial vehicles," *Defence Sci. J.*, vol. 66, no. 6, p. 651, Oct. 2016.

[55] A. J. Berry, J. Howitt, D.-W. Gu, and I. Postlethwaite, "A continuous local motion planning framework for unmanned vehicles in complex environments," *J. Intell. Robot. Syst.*, vol. 66, no. 4, pp. 477–494, Jun. 2012.

[56] Z. Yang, Z. Fang, and P. Li, "Decentralized 4D trajectory generation for UAVs based on improved intrinsic tau guidance strategy," *Int. J. Adv. Robot. Syst.*, vol. 13, no. 3, p. 88, May 2016.

[57] S. Lim and H. Bang, "UAV guidance laws to arrival at desired position and time from desired direction," in *Proc. 11th Int. Conf. Control, Autom. Syst.*, Oct. 2011, pp. 299–304.

[58] Z. Yang, C. Li, and G. Liu, "Cooperative 4D guidance for multiple UAVs based on tensor field," in *Proc. 13th World Congr. Intell. Control Autom. (WCICA)*, Jul. 2018, pp. 1709–1714.

[59] N. E. Dougui, D. Delahaye, S. Puechmorel, and M. Mongeau, "A new method for generating optimal conflict free 4D trajectory," in *Proc. 4th Int. Conf. Res. Air Transp. (ICRAT)*, 2010, pp. pp–185.

[60] N. Dougui, D. Delahaye, S. Puechmorel, and M. Mongeau, "A light-propagation model for aircraft trajectory planning," *J. Global Optim.*, vol. 56, no. 3, pp. 873–895, Jul. 2013.

[61] A. Ziggelaar, "How did the wave theory of light take shape in the mind of Christiaan Huygens?" *Ann. Sci.*, vol. 37, no. 2, pp. 179–187, Mar. 1980.

[62] J. Carballeira, C. Nicolás, S. Garrido, and L. Moreno, "Wildfire spreading simulator using fast marching algorithm," *SNE Simul. Notes Eur.*, vol. 31, no. 3, pp. 159–167, 2021.

[63] J. V. G. González, "Fast marching methods in path and motion planning: Improvements and high-level applications," Ph.D. dissertation, Dept. Syst. Eng. Automatics, Univ. Carlos III de Madrid, Madrid, Spain, 2015.

[64] J. A. Sethian, "A fast marching level set method for monotonically advancing fronts," *Proc. Nat. Acad. Sci. USA*, vol. 93, no. 4, pp. 1591–1595, Feb. 1996.

[65] A. Valero-Gomez, J. V. Gomez, S. Garrido, and L. Moreno, "The path to efficiency: Fast marching method for safer, more efficient mobile robot trajectories," *IEEE Robot. Autom. Mag.*, vol. 20, no. 4, pp. 111–120, Dec. 2013.

[66] B. L. Palomino, J. M. Mendi, and L. M. Lorente, "Planificacion y manejo de conflictos basado en fast marching square para uavs en entor, nos 3D de Grandes dimensiones," in *Proc. 43rd Jornadas de Automática*. A Coruña, Spain: Universidade da Coruña. Servizo de Publicacións, 2022, pp. 735–742.

[67] L. Yatziv, A. Bartesaghi, and G. Sapiro, "O(N) implementation of the fast marching algorithm," *J. Comput. Phys.*, vol. 212, no. 2, pp. 393–399, 2006.

[68] J. Muñoz, B. López, F. Quevedo, R. Barber, S. Garrido, and L. Moreno, "Geometrically constrained path planning for robotic grasping with differential evolution and fast marching square," *Robotica*, vol. 41, no. 2, pp. 414–432, Feb. 2023.

[69] S. Garrido, J. Muñoz, B. López, F. Quevedo, C. A. Monje, and L. Moreno, "Fast marching techniques for teaming UAV's applications in complex terrain," *Drones*, vol. 7, no. 2, p. 84, Jan. 2023.

[70] J. V. Gómez, D. Álvarez, S. Garrido, and L. Moreno, "Fast methods for eikonal equations: An experimental survey," *IEEE Access*, vol. 7, pp. 39005–39029, 2019.

[71] G. Tan, J. Zou, J. Zhuang, L. Wan, H. Sun, and Z. Sun, "Fast marching square method based intelligent navigation of the unmanned surface vehicle swarm in restricted waters," *Appl. Ocean Res.*, vol. 95, Feb. 2020, Art. no. 102018.

[72] K. Mrudul, R. K. Mandava, and P. R. Vundavilli, "An efficient path planning algorithm for biped robot using fast marching method," *Proc. Comput. Sci.*, vol. 133, pp. 116–123, Jan. 2018.

[73] Y. Liu, R. Song, R. Bucknall, and X. Zhang, "Intelligent multi-task allocation and planning for multiple unmanned surface vehicles (USVs) using self-organising maps and fast marching method," *Inf. Sci.*, vol. 496, pp. 180–197, Sep. 2019.

[74] P. Chen, Y. Huang, E. Papadimitriou, J. Mou, and P. van Gelder, "Global path planning for autonomous ship: A hybrid approach of fast marching square and velocity obstacles methods," *Ocean Eng.*, vol. 214, Oct. 2020, Art. no. 107793.

[75] J.-M. Mirebeau and J. Portegies, "Hamiltonian fast marching: A numerical solver for anisotropic and non-holonomic eikonal PDEs," *Image Process. Line*, vol. 9, pp. 47–93, Feb. 2019.

[76] D. Garcia, "Robust smoothing of gridded data in one and higher dimensions with missing values," *Comput. Statist. Data Anal.*, vol. 54, no. 4, pp. 1167–1178, Apr. 2010.

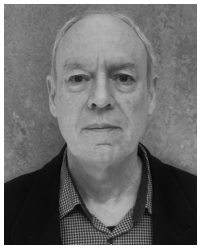
Blanca López received the degree in electronics and automation engineering from the School of Industrial and Aerospace Engineering, The University of Toledo, in 2019, and the M.S. degree in robotics and automation from the University Carlos III of Madrid, Spain, in 2021, where she is currently pursuing the Ph.D. degree with the Robotics Laboratory. Her research interests include path planning, coordination, and collision risk management strategies for multi-robot systems, mainly aimed at UAV teams and robotic manipulators.

Javier Muñoz was born in Zaragoza, Spain, in 1997. He received the B.S. degree in electronics and automation engineering from the School of Engineering and Architecture, Universidad de Zaragoza, in 2019, and the M.S. degree in robotics and automation from the University Carlos III of Madrid, Spain, in 2021, where he is currently pursuing the Ph.D. degree with the Robotics Laboratory. His research interests include Gaussian processes, Bayesian learning, path planning, UAVs, and grasp planning.

Fernando Quevedo (Graduate Student Member, IEEE) was born in Palma de Mallorca, Spain, in 1997. He received the B.S. degree in bioengineering from Universidad de Málaga in 2019. He is currently pursuing the master's degree in robotics and automatization with the University Carlos III of Madrid. He is collaborating with the Humasoft project researching models for soft robotics, 4D path planners, and network control schemes for multi-UAV systems.

Concepción A. Monje (Member, IEEE) received the M.Sc. degree in electronics engineering from the School of Industrial Engineering, University of Extremadura, Spain, in 2001, and the Ph.D. degree in industrial engineering from the University of Extremadura in 2006. In 2006, she joined the University Carlos III of Madrid, Spain, where she works in the area of fractional order control and robotics. She has been involved in many international and national research and industrial projects and published more than 130 research papers in high-impact conferences and journals.

Santiago Garrido (Member, IEEE) received the degree in physics from the University Carlos III of Madrid (UC3M), Spain, in 1955, the degree in mathematics from the Complutense University of Madrid, Spain, in 1979, and the Ph.D. degree from UC3M in 2000. His Ph.D. work studied the identification, estimation, and control of nonlinear systems. In 1997, he joined the Department of Systems Engineering and Automation, UC3M. His research interests include mobile robotics and manipulators, environment modeling, path planning, mobile robot global localization, and SLAM.



Luis Moreno (Member, IEEE) received the degree in automation and electronics engineering and the Ph.D. degree from Universidad Politécnica de Madrid, Spain, in 1984 and 1988, respectively. In 1994, he joined the Department of Systems Engineering and Automation, University Carlos III of Madrid, Spain, where he has been involved in many mobile robotics projects. His research interests include mobile robotics and manipulators, environment modeling, path planning, and mobile robot global localization problems.