Fast Marching Tree applied to geodesic trajectories in presence of wind: a day of flights in Europe study

Céline Demouge, Andréas Guitart, Daniel Delahaye

Fast Marching Tree applied to geodesic trajectories in presence of wind: a day of flights in Europe study

Céline Demouge, Andréas Guitart, and Daniel Delahaye.

***Abstract*—This paper addresses the generation of aircraft cruise optimal trajectories in the presence of wind using FMT* algorithm. After presenting the problem handled, the adaptation of the algorithm to solve it are presented. In particular, how to deal with great circle distances without increasing too much the computation time is detailed. The proposed method is evaluated on benchmarks and is shown as appropriate to solve this problem. A case study on European flights is also given. This case study shows that this method obtains good results on most European origin-destination pairs. This work is a preliminary work to show that FMT* can be used for aircraft cruise trajectories planning in the presence of wind.**

***Index Terms*—Cruise trajectory, Sampling-based path planning algorithm, Fast marching tree, Optimal aircraft trajectory, Wind.**

I. Introduction

Air transport has seen many technical and technological evolutions over the last few decades, enabling a high level of safety to be achieved while allowing significant growth. Aircraft and ground systems (tools for air traffic controllers, airlines, airport operators, etc.) have been the subjects of these developments, which have made air travel the safest mode of transportation in the world, and in a rapidly expanding market.

For this growth to continue, the air transport industry must now turn its attention to the environment. It must continue to guarantee a high level of safety and be able to accept traffic growth while reducing its environmental impact in a sustainable development approach. This approach is essential to ensure the sustainability of the sector and is encouraged by the authorities.

The Covid-19 crisis threatens the growth of air traffic and weakens the whole sector. Air transport actors must reinvent their ways of acting in order to recover the past demand. Today, the public is convinced of air transport safety, which must now prove its ability to meet current and future challenges. One of them is to decrease the environmental impact of the sector, without decreasing safety and capacity levels. Thus, measures yielding to save fuel are necessary. Moreover, in these times of crisis, such measures also allow financial savings since fuel is a major source of cost for airlines.

In the major part of the flight, in the cruise phase, the best operational mean to save fuel is to take benefit from wind. This paper presents a new method to optimize this phase of flight by taking into account the wind. For this purpose, a new method, initially issued from the robotics field and being the object of some studies in aeronautics is used: the Fast Marching Tree (FMT*) algorithm. Several issues have to be considered: it is necessary to take into account great circle distances and to take into account a cost different from the distance that is traditionally the one used when sampling based methods are implemented.

This paper is organized as follows. Section II presents different previous related works on optimal cruise trajectory computation in the presence of wind and sampling based methods. Then, in Section III the problem under study is detailed before explaining the method used to solve it in Section IV. Finally, some results from benchmarks and from case studies are shown in Section V and in Section VI respectively.

C. Demouge, A. Guitart, and D. Delahaye are with ENAC, Université de Toulouse, 7 avenue Edouard Belin, Toulouse, 31400 France (email: firstname.surname@enac.fr).

II. Previous related works

This section presents some previous related works on the computation of aircraft optimal cruise trajectory in the presence of wind, before focusing on a particular type of path planning method: sampling-based methods.

A. Wind optimal cruise trajectory computation

1) Optimal control approach: The problem under study requires finding an optimal trajectory for an aircraft. This problem can, in the most intuitive way, be modelized as an optimal control problem thanks to the **equations of the flight dynamics**. The cost to be minimized can be the flight time or the fuel consumption for instance. Several authors have modeled the problem of optimal trajectories in the presence of wind for one aircraft in this form. For instance, Sridhar *et al.* [1], or Palopo *et al.* [2] who used a method from Jardin and Bryson [3].

The last article cited [3] mentions the **Zermelo problem** [4] which was proposed in 1931. The aim of this problem is to optimize the navigation of a boat to take the best advantage of the marine currents, a problem that is similar to the problem addressed in this paper. Thus, it is often considered as a reference benchmark for methods implemented in the literature.

2) Graph-based approach: Several methods choose instead to discretize the environment and work on the resulting graph, making the problem simpler and more similar to **well-known shortest path problems** or tree search. This graph is built

from a discretization of the space. The cost of edges depends on the great circle distances and takes also into account the strength of the wind in the direction of the edges.

Different algorithms can then be used such as **Dijkstra** [5] or **A*** [6]. These methods can be very efficient but several problems arise:

- The trajectory created is sub-optimal since the search space is restricted because it is discretized. Indeed, the trajectory found is optimal regarding the graph but not considering the whole space;
- The trajectory created is not directly "flyable". The trajectory does not follow the requirements of aircraft dynamics. Then, additional work for smoothing the trajectory is required, and therefore the optimum may move further away.

3) Propagation methods: From the analogy of light propagation in an open environment and the Eikonal equation governing this propagation or other Hamilton-Jacobi type equations, very efficient methods have been created, called propagation methods. The principle of these methods is therefore **to propagate a "wave"** (or equivalent front) on a grid and then **back propagate the gradients obtained on this grid to draw a continuous trajectory**.

Fast marching [7] and ordered upwind [8] methods are based on an older method developed by Sethian called level set method [9]. These methods were developed to solve stationary Hamilton-Jacobi equations, *i.e.* equations of the type :

$$||\nabla u(X)|| = f(X, \frac{\nabla u(X)}{||\nabla u(X)||}) \tag{1}$$

where X is the state vector, u is the optimal cost representing the minimum arrival time, and f is the "slowness" of the domain at any point, for example the inverse of the propagation speed of the front.

Fast marching methods are particularly adapted for isotropic propagation. However, in the case of wind, an anisotropic and adapted method has been developed, for instance by Mirebeau *et al.* [10]. Another method, simpler and already proven to be efficient, the **ordered upwind** method, is presented in [8]. It has been used in the PhD thesis [11] in the 2D case.

B. Sampling based methods

Another type of path planning methods is sampling based methods. They are non-deterministic discrete methods. They take advantage of random sampling to improve and speed up the search for the shortest or lowest cost path.

1) General principles and main algorithms: Research on this type of methods was first driven by robotics. The goal was initially to find a path, as short as possible, for a robot, avoiding obstacles. When graph methods are used, to improve the solution obtained, graphs can be densified but this often results in a very high computational cost. To overcome these issues, some methods have been developed to **sample the environment**, first by more or less randomly choosing a set of points, see for instance Simeon *et al.* [12] or Kavraki *et al.* [13]. These methods, named **Probabilistic RoadMaps (PRM)**, are based on two steps:

1) Creation of a graph from the sampling of the environment;
2) Finding the shortest path from one point to another.

PRM is based on random sampling and this can be problematic regarding the solution optimality in the whole space and the variance from a run to another. To avoid relying too much on random sampling, other **sampling methods** have been developed, particularly **Rapidly-exploring Random Tree (RRT)** [14].

The goal of this type of algorithm is also to reduce the number of points in the environment, and to find the most interesting ones by **creating a graph in an efficient way**, to recover a maximum of useful information on the environment in order to find a path close to the optimal one while saving a lot of time. **RRT expands naturally into unexplored areas**. It is this property based on a natural bias, called Voronoï bias, that makes this method effective.

RRT has then been improved into **RRT*** [15] which has been proven to be **asymptotically optimal**, unlike RRT.

Another sampling based method is the **Fast Marching Tree (FMT*)** [16]. This method executes concurentially, and thus "at the same time", the two steps of RRT*. Indeed, RRT* first builds the graph and then tries to optimize locally the connections, while FMT* builds the graph by choosing the locally optimal connection. This saves a lot of time. Another idea is added to save even more time: FMT* performs a dynamic "lazy" computation, *i.e.* the algorithm initially ignores obstacles and if a locally-optimal edge is created and crosses an obstacle, then it ignores it. This method is detailed in Section IV.

2) Application to aeronautics: Sampling based methods have recently been adapted for certain problems arising from air transport, drones or other aerial vehicles.

RRT* has been used for aeronautical applications. In 2017, Pharpatara *et al.* [17] used RRT* to compute obstacle avoidance trajectories in 3D. This paper therefore shows that RRT* is easily usable in 3D. However, the authors consider the wind to be negligible given the speed of the aircraft. So their article is not directly relevant to our problem, although it does give a lead to the study of obstacle avoidance and shows that the use of these methods for aircraft trajectories can be interesting. The 3D wind track using RRT was however studied earlier, for glider-type drones, by Chakrabarty and Langelaan [18]. In their paper, encouraging and interesting results are shown for maximizing the glider's flight time or getting it to reach a given point by using air currents. RRT has also been used for emergency aircraft trajectory design [19].

FMT* has also been used for aeronautical applications. It has for instance been used for aircraft emergency trajectory design [20]. It has been chosen for its little computation time since it is a key point for this problem. It has also been used for aerial vehicles with complex dynamics, for instance for soaring flight in [21].

To conclude, various methods have already been developed for the optimization of aircraft trajectories in a wind field. But new methods from robotics are being developed in air transport. The study presented here proposes to use the FMT* to solve this problem.

III. Mathematical modeling

This section deals with the mathematical modeling of our problem. First, the search space and context are detailed in Subsection III-A while the objective function is detailed in Subsection III-B. Then, constraints are presented in Subsection III-C.

A. Search space

The trajectories are computed in the context of *Free Flight* [22]. Initiated to improve Air Traffic Management, it is based on the notion of 4D trajectory planning (space and time). Airspace users are allowed to select their route without any constraints, using favorable winds and avoiding hazards.

In this context, the search space of the problem is the 4D-space (3D and time) without any constraint of predefined routes. In this study, no weather hazard is considered and since the cruise phase is under study, no ground obstacles are considered, nor thunderstorms. Moreover, the study focuses on only one aircraft, so no other traffic is considered.

B. Objective function

The first goal of the study presented here is to minimize fuel consumption for the cruise phase of an aircraft in the context explained before. This background allows to make several hypotheses:

- The optimization is done in a 2D plane (the altitude is constant).
- The True AirSpeed (TAS) is constant.

These assumptions lead to a simplification of the problem. Indeed, they lead to accepting the fact that the fuel flow is constant. In this case, minimizing fuel consumption is equivalent to minimizing flight time between the departure point and the arrival point.

The objective function to be minimized is then written in Equation 2.

$$f(\gamma) = \int_{t_0}^{t_f} dt = t_f - t_0 \tag{2}$$

where γ is the computed trajectory, t_0 the departure time and t_f the arrival time.

Usually in aeronautics, 2D or 3D navigation points called waypoints are used to define a trajectory. In this study, γ is considered to be a sequence of N 2D-waypoints $\{w_i\}_{0 \leq i \leq N}$. The objective function can be rewritten in a discretized sum of flight times as shown by Equation 3.

$$f(\gamma) = g(w_1, ..., w_N) = \sum_{i=1}^{N-1} \Delta t_{i,i+1} \tag{3}$$

where $\Delta t_{i,i+1}$ is the flight time between the two points w_i and w_{i+1}. It is defined by Equation 4.

$$\Delta t_{i,i+1} = \frac{d_{i,i+1}}{GS_{i,i+1}} \tag{4}$$

where $d_{i,i+1}$ is the distance between w_i and w_{i+1} and $GS_{i,i+1}$ is the ground speed between these two points. The ground speed is defined as follows:

$$GS = \|\mathbf{GS}\| = \|\mathbf{TAS} + \mathbf{W}\| \tag{5}$$

where $\mathbf{TAS}$ is the True AirSpeed of the aircraft and $\mathbf{W}$ the wind encountered. Figure 1 shows a graphical representation of this equation.

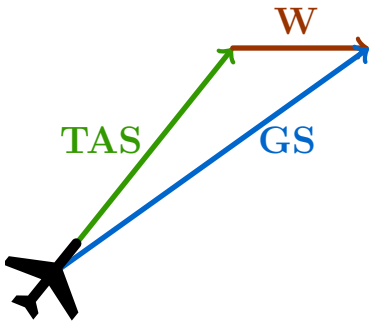


Fig. 1: Graphical representation of the speed triangle related to Equation 5.

On a sphere, the shortest path between two points in terms of distance is the great circle. Figure 2 shows the difference between the straight line and the great circle.

Fig. 2: Comparison between the straight line and the great circle trajectory between Paris and New York.

Great circle distance can be computed thanks to Equation 6.

$$\begin{aligned} d_{i,i+1} = 60 \arccos\big(&\sin(\phi_i)\sin(\phi_{i+1}) \\ &+ \cos(\phi_i)\cos(\phi_{i+1})\cos(\lambda_{i+1} - \lambda_i)\big) \end{aligned} \tag{6}$$

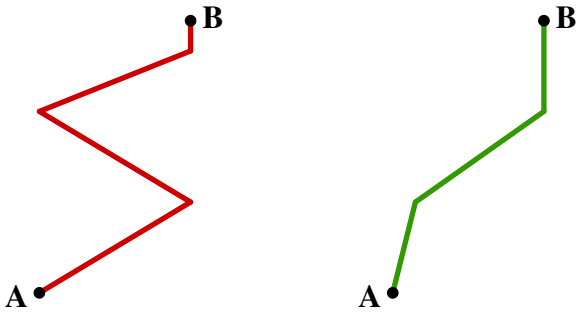
where λ_i (resp. λ_{i+1}) is the longitude of point w_i (resp. w_{i+1}) and ϕ_i (resp. ϕ_{i+1}) is the latitude of point w_i (resp. w_{i+1}). $d_{i,i+1}$ is expressed in Nautical Miles (NM).

C. Constraints

The main constraints linked to the problem under study is to compute a "flyable" trajectory. In the case of this study, the trajectory is computed in the 2D plane, that is to say only lateral constraints are taken into account for the definition of a "flyable" trajectory.

The constraints for the flyability are in general ensuring the flight dynamics equations. Nevertheless, in the case of this

study, they will be limited to the fact that the turns should not be too sharp. Figure 3 shows, for example, the type of trajectory that we want to avoid and the one that will be preferred instead.



(a) Trajectory with sharp turns

(b) Trajectory without sharp turns

Fig. 3: Trajectory from a point A to a point B with or without sharp turns.

IV. Path optimization method

A. FMT* algorithm

Fast Marching Tree (FMT*) has been introduced by Lucas Janson, Edward Schmerling, Ashley Clark and Marco Pavone in [16] in 2015. FMT* is proven to be asymptotically optimal and converge faster than RRT*. The sequel of this subsection describes the algorithm.

First, the environment is (randomly) sampled with N points or nodes. Then, FMT* will explore each node to compute the optimal path from the origin to the goal. For this, at each iteration, nodes are split into 3 sets:

1) $V_{unvisited}$: nodes that have never been visited by the FMT* (in green in the following figures);
2) V_{open}: nodes that have already been visited but for which the cost value is not definitively computed (in orange in the following figures);
3) V_{closed}: nodes that have already been visited but for which the cost value is definitively computed (in red in the following figures).

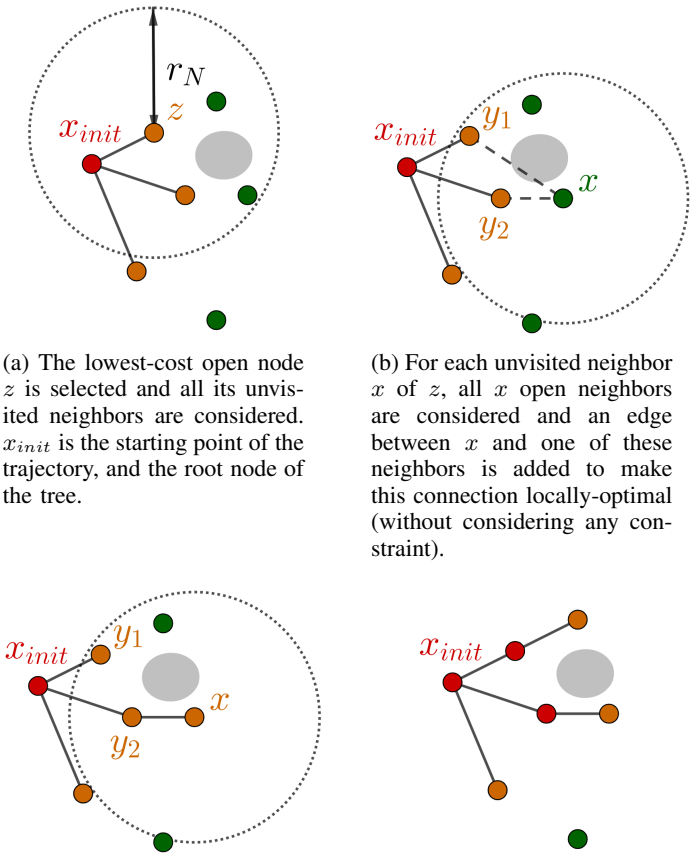
Figure 4 details one iteration.

All the nodes at a distance r_N of a node are considered as its neighbors. r_N is the neighborhood radius and is a function of the number N of samplings. The radius used in [16] is:

$$r_N = (1+\eta)\ r_N^* \tag{7}$$

where $\eta > 0$ and r_N^* is the minimum neighboring radius usable defined by:

$$r_N^* = 2\ \left(\frac{1}{d}\right)^{1/d}\ \left(\frac{\mu(\chi_{free})}{\zeta_d}\right)^{1/d}\ \left(\frac{\log(N)}{N}\right)^{1/d} \tag{8}$$



(a) The lowest-cost open node z is selected and all its unvisited neighbors are considered. x_{init} is the starting point of the trajectory, and the root node of the tree.

(b) For each unvisited neighbor x of z, all x open neighbors are considered and an edge between x and one of these neighbors is added to make this connection locally-optimal (without considering any constraint).

(c) If this connection does not violate constraints or is collision-free, it is added to the tree and x is removed from the set $V_{unvisited}$ and added to V_{open}.

(d) Once all z unvisited neighbors have been visited, z is put in V_{closed} and the iteration is done.

Fig. 4: Details of one iteration of FMT*.

where d is the space dimension, χ_{free} the free space, $\mu(\chi_{free})$ its Lebesgue measure[1], ζ_d the volume of the d-dimensional unitary ball[2].

Algorithm 1 is the complete FMT* algorithm in pseudo-code. The set of neighbor of a node x is noted $Neighbors(x)$.

To ensure the lowest possible complexity, the set V_{open} is implemented as a priority queue which allows an element to be extracted (line 6) in a constant time and an element to be added in $O(\log n)$ (with n the size of the queue).

B. Aeronautical constraints

As explained in Subsection III-C, some aeronautical constraints should be taken into account and especially for turns. To ensure the flyability of the trajectory, first, the radius of neighborhood, r_N, can be increased in order to somehow smooth the computed trajectory even if the computation time

[1] $\mu([a1,b1]\times[a2,b2]) = (b1-a1)*(b2-a2)$ where $b1 > a1$ and $b2 > a2$.
$\mu([a1,b1]\times[a2,b2]\times[a3,b3]) = (b1-a1)*(b2-a2)*(b3-a3)$ where $b1 > a1$, $b2 > a2$ and $b3 > a3$.

[2] In 2D space, ζ_2 is the surface of a disk of radius 1 ($\zeta_2 = \pi$).
In 3D space, ζ_3 is the volume of a ball of radius 1 ($\zeta_3 = \frac{4}{3}\pi$).

Algorithm 1: FMT* algorithm.

Input: x_{init}, sample set V (with $x_{init} \in V$), χ_{goal} $(\exists x \in \chi_{goal}, x \in V)$
Output: *succeed*

$succeed \leftarrow false$
$V_{open} \leftarrow \{x_{init}\}$
$V_{unvisited} \leftarrow V \backslash \{x_{init}\}$
$Tree$ initialized with its root x_{init}
while $V_{open} \neq \emptyset \wedge \neg$ *succeed* **do**
  $z \leftarrow$ lowest-cost node in V_{open}
  $V_{successfully\_connected} \leftarrow \emptyset$
  for $x \in V_{unvisited} \cap Neighbors(z)$ **do**
    $e \leftarrow$ locally-optimal edge to x from a node in $V_{open} \cap Neighbors(x)$
    if e *is collision-free* **then**
      $Tree \leftarrow Tree \cup e$
      $V_{successfully\_connected} \leftarrow V_{successfully\_connected} \cup \{x\}$
  $V_{unvisited} \leftarrow V_{unvisited} \backslash V_{successfully\_connected}$
  $V_{open} \leftarrow V_{open} \cup V_{successfully\_connected}$
  $V_{open} \leftarrow V_{open} \backslash \{z\}$
  $V_{closed} \leftarrow V_{closed} \cup \{z\}$
  if $z \in \chi_{goal}$ **then**
    $succeed \leftarrow true$

is also increased. Since the computation time is not critical in this application, the radius, r_N, is typically multiplied by 10 in respect to the optimal radius.

Second, the result of FMT* is actually a sequence of waypoints. It can be directly given to the aircraft FMS for computing the trajectory. This process is detailed in Figure 5.

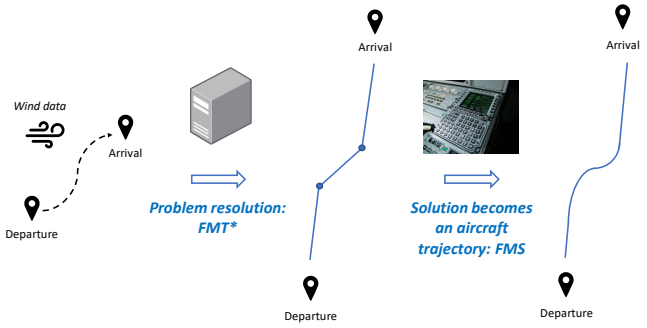


Fig. 5: Aeronautical constraints: FMT* waypoints list inserted in the FMS.

C. Taking into account great circle distances

As explained before, it is necessary to take into account great circle distances for the computation of costs. In that case, it is necessary to define which distance is used at each step.

When looking for neighbors, the distance considered is the Euclidean distance between two points. Thus, the coordinates of the points will be taken into account to establish this distance. This is the same as calculating a Euclidean distance on a Mercartor type map, like the one shown in Figure 6.

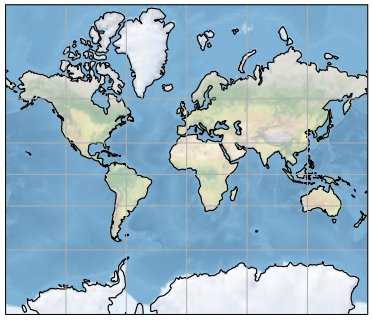


Fig. 6: Example of a map with Mercator projection.

When computing the cost, and thus finding the locally optimal neighbor of a point, the "distance" used is the great circle distance as detailed in Subsection III-B. This distinction is essential in the case of application here. Indeed, when this algorithm is applied to robotics, only one distance is involved. Often, the cost is even equal to the distance when trying to find the shortest path avoiding obstacles. Here two different distances are involved, and it is therefore necessary to strictly differentiate the cost from the Euclidean distance used for the neighborhoods.

As shown by Equation 6, great circle distance can be time-consuming to compute. From this observation, another method that avoid this pitfall should be developed. This other method is based on the assumption that between two nearby points, typically one in the neighborhood of the other in the sense of the algorithm, the great circle can be approximated by the straight line. Figure 7 shows an example of the validity of this approximation.

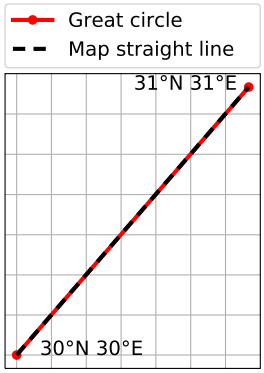


Fig. 7: Straight line and great circle trajectory between two near points.

Once this approximation done between two neighbors, a coefficient must be applied to approximate the earth's curvature as shown by the Equation 9 between two points $A(\phi_A, \lambda_A)$ and $B(\phi_B, \lambda_B)$.

$$d_{A,B} \approx \sqrt{\left(\frac{\phi_B - \phi_A}{m_\phi}\right)^2 + \left(\frac{\lambda_B - \lambda_A}{m_\lambda}\right)^2} \tag{9}$$

where the coefficients m_λ and m_ϕ are computed according to the following principle:

$$m = \frac{\text{distance on the Cartesian plane} \approx \text{distance at the equator}}{\text{distance on the Earth}} \tag{10}$$

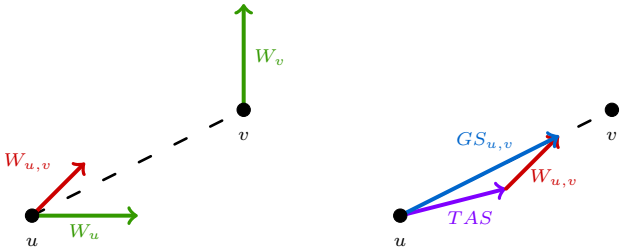
Then m_ϕ and m_λ can be computed at each sampled node thanks to Equations 11 and 12 with accuracy h. For longitude, there is no need to differentiate east and west since the result will be the same. For latitude, the plus sign is used if $\phi_A < \phi_B$ (if B is in the north of A) else the minus sign is used.

$$\begin{aligned} m_\lambda &= \frac{d_{(0,0)(0,h)}}{d_{(\phi_A,\lambda_A)(\phi_A,\lambda_A+h)}} \\ &= \frac{h}{\arccos\left(\sin^2(\phi_A) + \cos^2(\phi_A)\cos(h)\right)} \end{aligned} \tag{11}$$

$$\begin{aligned} m_\phi &= \frac{d_{(0,0)(\pm h,0)}}{d_{(\phi_A,\lambda_A)(\phi_A \pm h,\lambda_A)}} \\ &= \frac{h}{\arccos\left(\sin(\phi_A)\sin(\phi_A \pm h) + \cos(\phi_A)\cos(\phi_A \pm h)\right)} \end{aligned} \tag{12}$$

D. Taking into account wind

The wind is taken into account in the cost computation with the great circle distance. Typically, it comes in when the locally optimal connection is searched in the algorithm. The ground speed between two nodes u and v is computed thanks to the mean wind between two nodes $W_{u,v} = \frac{W_u + W_v}{2}$ and is directed on the edge as shown in Figure 8. The true airspeed of the aircraft is considered constant all along the trajectory.



(a) The wind taken into account between two nodes is the mean wind.

(b) Ground speed computation between two nodes u and v.

Fig. 8: Computing the ground speed taking into account wind between two nodes u and v.

Then the cost $c_{u,v}$ computed between a node u and a node v is:

$$c_{u,v} = \frac{d_{u,v}}{GS_{u,v}} \tag{13}$$

where $d_{u,v}$ is computed thanks to Equation 9 and $GS_{u,v}$ is computed as shown by Figure 8.

V. Results on academic benchmarks

This section shows the performances of the developed method on two benchmarks. First, the method is evaluated in Subsection V-A on Zermelo's benchmark to evaluate the consideration of wind. Then, the method is evaluated in Subsection V-B on a great circle trajectory computation without wind. In the sequel, the various results shown are obtained with a computer with an Intel Core i5-10210U, 1.60 Hz, with 8 Go RAM and a Debian Linux OS. The implementation is done in Java.

A. Zermelo's benchmark

As mentioned in the state of the art, a benchmark problem that can be used to evaluate algorithms computing optimal trajectories in the presence of wind is the Zermelo's problem. It was first proposed in 1931 by Ernst Zermelo in [4]. The goal of this problem is to minimize the travel time from point A to point B of a boat subject to water currents, which can easily be assimilated to the wind in the case of our application and the boat to an aircraft. In some particular cases, this problem has closed form solution. In the following, a particular case of wind will be studied: the **case of a linear wind**.

The problem is treated using **Cartesian coordinates**. Let $W(x, y)$ be the wind vector.

$$\forall (x, y),\ W(x, y) = \begin{pmatrix} \frac{V_a}{4}\, y \\ 0 \end{pmatrix} \tag{14}$$

where V_a is the airspeed of the aircraft.

The value $\frac{V_a}{4}$ was chosen because it is a classical value of relatively strong wind at altitude.

Figure 9 shows the wind encountered in this setup.

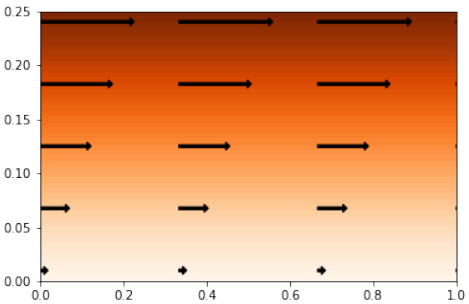


Fig. 9: Linear wind case for Zermelo problem.

Figure 10 and Table I show results obtained from FMT* to compute the solution of Zermelo problem in this setup. To obtain such smooth trajectories, the chosen neighborhood radius is larger than the minimum one, r_N^* defined by Equation 8, it has been multiplied by 5 to obtain the following results.

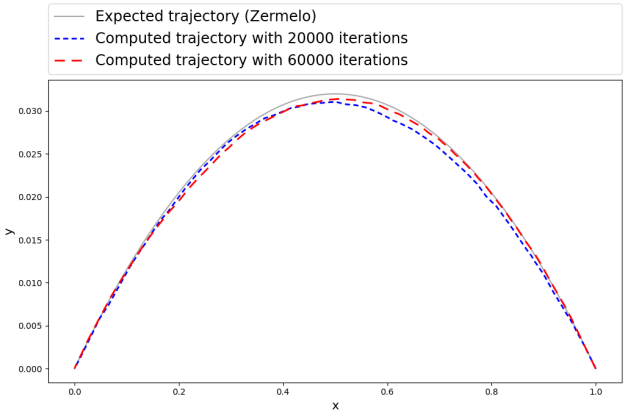


Fig. 10: Results obtained from FMT* - Zermelo problem in the case of linear wind with different maximum of iterations.

TABLE I: Relative errors and execution times for computing a solution to Zermelo problem with FMT*.

Iterations number	Distance relative error (%)	Time relative error (%)	Computation time (s)
10000	0.01995	0.02560	0.36
20000	0.01954	0.01542	1.64
30000	0.01764	0.01537	4.11
40000	0.01313	0.01534	8.66
50000	0.01462	0.01533	16.63
60000	0.01063	0.01532	25.35

B. Computing great circle

Moreover, the method has been tested in a windless case to evaluate the capacity of the algorithm to compute great circle trajectory between two points. Figure 11 and Table II show results obtained from FMT* to compute the great circle trajectory between two points with different maximum number of iterations.

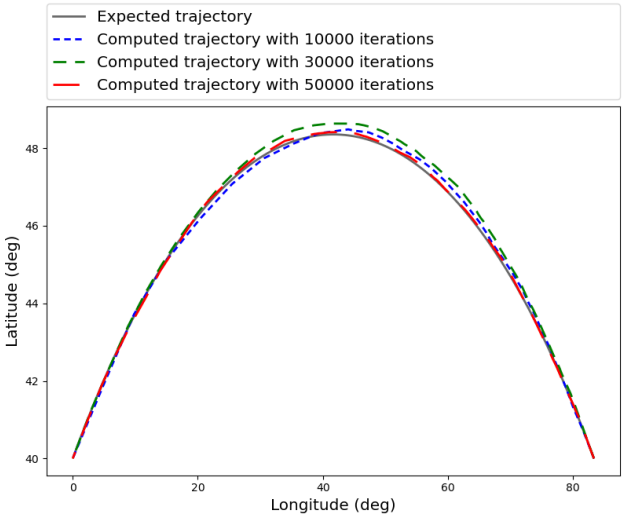


Fig. 11: Results obtained from FMT* - Computation of the great circle trajectory between two points with different maximum of iterations.

TABLE II: Relative errors and execution times for great circle trajectory computation with FMT*.

Iterations number	Relative error (%)	Computation time (s)
10000	0.43602	0.59
20000	0.42486	1.51
30000	0.43225	5.03
40000	0.43242	9.57
50000	0.42232	17.37
60000	0.42113	26.19

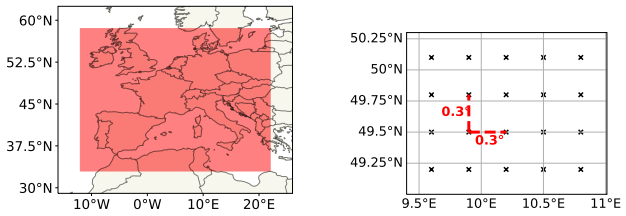
In that case, the error does not always decrease since there is an approximation on the great circle distance.

VI. Case Study

This section presents a case study on most flown European flights. First, Subsection VI-A details the weather data used, then Subsection VI-B details how flights are selected, then, in Subsection VI-C, results are presented.

A. Weather data

Weather data are extracted from the API of the Windy website [23]. The day chosen is October 14, 2022 in the morning, at Flight Level 390 (FL 390), that is to say at altitude-pressure 200 hPa. Wind data have been extracted on a 2D-grid with a resolution of 0.3 degrees on latitude and longitude, from latitude 33 to latitude 58.5 and from longitude -12 to longitude 22. In terms of distance, the resolution is near to 18 NM (33 km) in latitude and 15 NM (28 km) in longitude (see Figure 12).



(a) Data extraction area. (b) Data extraction resolution.

Fig. 12: Wind data extraction features.

Data are then extrapolated on each point of the sampling using quadratic interpolations:

$$W(P) = \frac{\sum_{i=1}^{4} W(P_i)\, d_i^{-2}}{\sum_{i=1}^{4} d_i^{-2}} \tag{15}$$

where P_1, P_2, P_3 and P_4 are the grid points all around the sampling point P and d_1, d_2, d_3 and d_4 are the distances from P to these points as shown by Figure 13.

From this extraction, the following wind map is obtained:

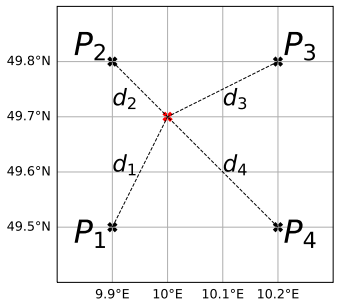


Fig. 13: Notations for quadratic interpolation of wind.

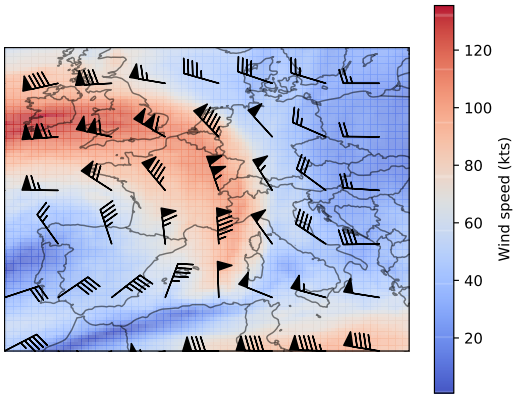
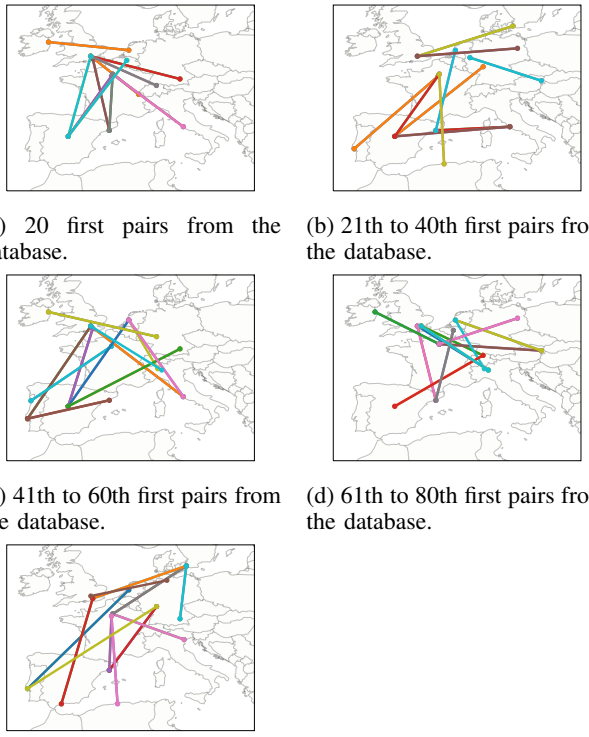


Fig. 14: Wind extracted for the study case.



(a) 20 first pairs from the database.

(b) 21th to 40th first pairs from the database.

(c) 41th to 60th first pairs from the database.

(d) 61th to 80th first pairs from the database.

(e) 81th to 100th first pairs from the database.

Fig. 15: Origin-destination pairs inputs for case study. Only the half of them are represented since if a pair is considered, it symmetric is also considered.

B. Flight data

To evaluate our method on real cases, it has been chosen to test it on the most flown origin-destination pair in Europe. For this, Eurocontrol R&D data [24] are used.

Data from March 2019 have been processed and the international flights longer than 400 NM (around 1 hour of flight without wind) have been selected. The 100 most flown pairs have been kept. The origins and destination are represented in Figure 15.

It is considered that flights are done by standard commercial aircraft and so the airspeed is considered constant and equal to 400kts. It is based on the cruise speed of an A320 [25].

C. Results

This subsection presents the results obtained from the detailed case study. First, some examples of computed trajectories are given then a summary of the gains obtained is presented. All results are obtained with 50,000 iterations of the FMT* algorithm. The first example presented is the trajectory from Amsterdam Schiphol airport (EHAM) to Lisbon airport (LPPT). It is the 91th origin-destination pair from the 100 flights database. The gain in flight time is 2.17 %, that is to say 3 minutes and 20 seconds. For this, the flight distance is increased by 1.22 %, that is to say 22.39 NM. It has been computed in 22.5 s.

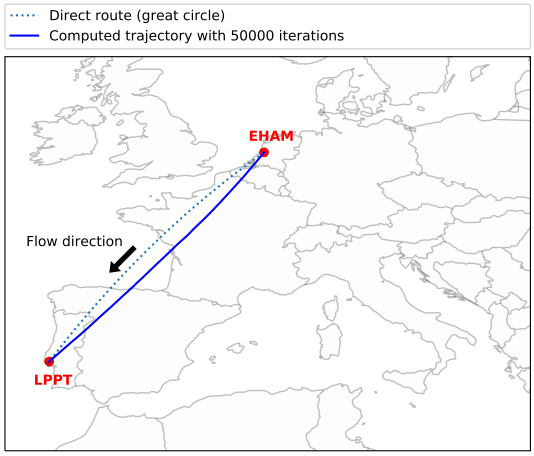


Fig. 16: Trajectory from Amsterdam Schiphol airport to Lisbon airport. The gain is 2.17 % in flight time with 1.22 % more flight distance.

The second example presented shows that sometimes it has no sense to deviate from the great circle trajectory. This example is the trajectory from Copenhagen airport (EKCH) to Munich airport (EDDM). It is the 100th origin-destination pair from the 100 flights database. Since the wind is weak between the origin and the destination, the gain is only 0.03 %, that is to say 1 second, which is not significant. It has been computed

in 35 s.

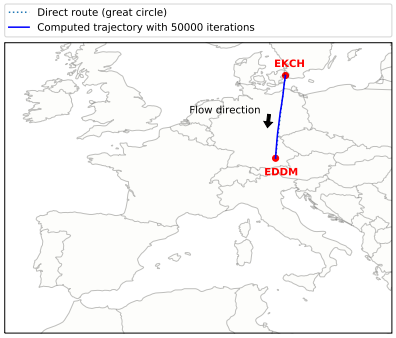


Fig. 17: Trajectory from Copenhagen airport to Munich airport. The gain is 0.03 %.

Let us summarize the different results obtained. Table III details the mean gain and the different quartiles of gain distribution. Table IV details the distance increase distribution. Figure 18 does the same in a graphical view thanks to box plots.

TABLE III: Flight time gain distribution on the case study.

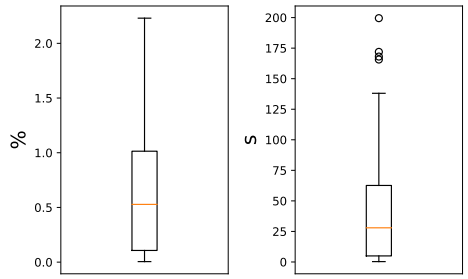
Mean	Q1	Median	Q3
43 s	28 s	5 s	63 s
0.66 %	0.53 %	0.11 %	1.01 %

TABLE IV: Distance increase distribution on the case study.

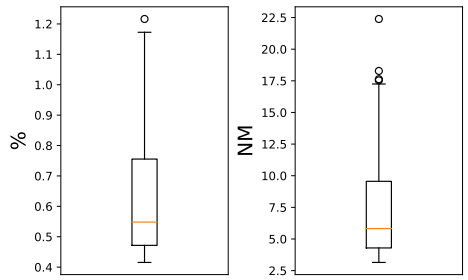
Mean	Q1	Median	Q3
7.35 NM	5.82 NM	4.29 NM	9.56 NM
0.63 %	0.55 %	0.47 %	0.76 %

This case study shows an interesting feature: as expected, the greater the distance between the origin and the destination is, the greater the benefit can be (see Figure 19).

This study gives comparable results to the ones found in literature. For instance, in [26], similar studies are done and for flight with a distance higher than 800 km for which flight times can be reduced between 25.99 s and 76.88 s on average. It shows that the method presented is efficient since it gives good results regarding the literature with little computation time.



(a) Flight time gain distribution in % (left) and in seconds (right).



(b) Distance flown increase distribution in % (left) and in nautical miles (right).

Fig. 18: Graphical view of flight time and flight distance distributions obtained on the study case.

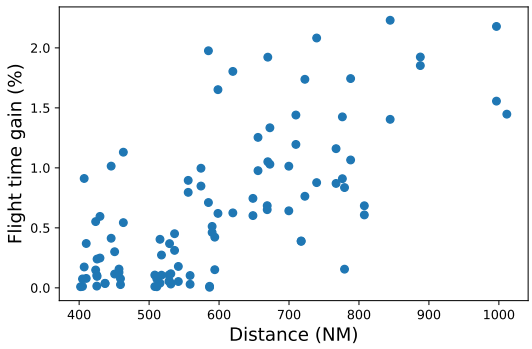


Fig. 19: Gain in flight time in function of distance between the origin and destination. The greater the distance between the origin and the destination is, the greater the benefit can be.

VII. Conclusion

This paper addresses a new application of the Fast Marching Tree (FMT*) algorithm. The literature shows some example of aeronautical applications of this algorithm originally made for robotics. This paper deals with the particular case of computing aircraft optimal trajectory in the presence of wind for cruise phase. After having presented different methods used for optimal trajectory computation in the presence of wind and sampling based methods in general, it presents the modelization of the problem. Then it details how it is solved. This part of the paper details how FMT* algorithm can be declined in the special case of cruise phase, by taking into account great circle distances and a cost function between two points which is not equal to the Cartesian distance between them. After having tested on benchmark problems, some study cases are presented. These study cases show that this algorithm is efficient that is to say that it provides solutions to save flight time as expected in a satisfactory computation time.

Some future works can be considered. First, this paper focuses on flight time minimization but the method described can be used with another cost function, for instance, fuel consumption or other climate impact functions. For instance, it can be adapted to the case of non-CO_2 effects (for instance

NO_x or contrails). Moreover, since obstacles are taken into account in the original algorithm, aeronautical obstacles can also be taken into account. For instance, dangerous weather phenomena such as thunderstorms can be avoided thanks to this method.

Another possible improvement of the presented method is the consideration of altitude. Even if during the cruise phase, the altitude is nearly constant, the cruise phase can be optimized and if they are really beneficial altitude changes could be considered. Finally, a multi-agent extension can be considered later to take into account several aircraft. This will allow optimizing all the trajectories by not creating conflicts between the planes and avoiding congestion in some airspaces.

REFERENCES

[1] B. Sridhar, H. K. Ng, and N. Y. Chen, "Aircraft trajectory optimization and contrails avoidance in the presence of winds," *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 5, pp. 1577–1584, Sep. 2011.

[2] K. Palopo, R. D. Windhorst, S. Suharwardy, and H.-T. Lee, "Wind-optimal routing in the national airspace system," *Journal of Aircraft*, vol. 47, no. 5, pp. 1584–1592, Sep. 2010.

[3] M. R. Jardin and A. E. Bryson, "Neighboring optimal aircraft guidance in winds," *Journal of Guidance, Control, and Dynamics*, vol. 24, no. 4, pp. 710–715, Jul. 2001.

[4] E. Zermelo, "Über das Navigationsproblem bei ruhender oder veränderlicher Windverteilung," *ZAMM - Zeitschrift für Angewandte Mathematik und Mechanik*, vol. 11, no. 2, pp. 114–124, 1931.

[5] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, Dec. 1959.

[6] P. Hart, N. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[7] J. A. Sethian, "A fast marching level set method for monotonically advancing fronts." *Proceedings of the National Academy of Sciences*, vol. 93, no. 4, pp. 1591–1595, Feb. 1996.

[8] J. A. Sethian and A. Vladimirsky, "Ordered upwind methods for static Hamilton–Jacobi equations," *Proceedings of the National Academy of Sciences*, vol. 98, no. 20, pp. 11 069–11 074, Sep. 2001.

[9] D. Adalsteinsson and J. A. Sethian, "A fast level set method for propagating interfaces," *Journal of Computational Physics*, vol. 118, no. 2, pp. 269–277, May 1995.

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

[11] B. Girardet, "Trafic aérien : détermination optimale et globale des trajectoires d'avion en présence de vent," PhD thesis, INSA de Toulouse, Dec. 2014.

[12] C. Nissoux, T. Simeon, and J.-P. Laumond, "Visibility based probabilistic roadmaps," in *Proceedings 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human and Environment Friendly Robots with High Intelligence and Emotional Quotients (Cat. No.99CH36289)*, vol. 3. Kyongju, South Korea: IEEE, 1999, pp. 1316–1321.

[13] L. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, Aug. 1996.

[14] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Ames, IA, USA*, 1998.

[15] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, Jun. 2011.

[16] L. Janson, E. Schmerling, A. Clark, and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," *The International Journal of Robotics Research*, vol. 34, no. 7, pp. 883–921, Jun. 2015.

[17] P. Pharpatara, B. Hérissé, and Y. Bestaoui, "3-D trajectory planning of aerial vehicles using RRT*," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 3, pp. 1116–1123, May 2017.

[18] A. Chakrabarty and J. Langelaan, "UAV flight path planning in time varying complex wind-fields," in *2013 American Control Conference*, Jun. 2013, pp. 2568–2574, iSSN: 2378-5861.

[19] R. Sáez, H. Khaledian, X. Prats, A. Guitart, D. Delahaye, and E. Feron, "A fast and flexible emergency trajectory generator enhancing emergency geometric planning with aircraft dynamics," Sep. 2021.

[20] A. Guitart, D. Delahaye, and E. Feron, "An accelerated dual fast marching tree applied to emergency geometric trajectory generation," *Aerospace*, vol. 9, no. 4, p. 180, Apr. 2022.

[21] L. Bonin, D. Delahaye, A. Guitart, E. Feron, and X. Prats, "Optimal path planning for soaring flight," in *Conference on Guidance Navigation and Control (CEAS EuroGNC 2022)*. Berlin, Germany: CEAS and AIAA, May 2022.

[22] R. Schultz, D. Shaner, and Y. Zhao, "Free flight concept," in *Guidance, Navigation, and Control Conference*, 1997, p. 3677.

[23] W. SE, "Windy API," 2022. [Online]. Available: https://api.windy.com/

[24] Eurocontrol, "R&D data," 2022. [Online]. Available: https://www.eurocontrol.int/dashboard/rnd-data-archive

[25] SKYbrary aviation safety, "Airbus A-320," 2022. [Online]. Available: https://www.skybrary.aero/aircraft/a320

[26] B. Girardet, L. Lapasset, D. Delahaye, and C. Rabut, "Wind-optimal path planning: Application to aircraft trajectories," in *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*, Dec. 2014, pp. 1403–1408.