CEAS-GNC-2022-032

CEAS EuroGNC 2022
"Conference on Guidance, Navigation and Control"
3-5 May 2022 @ Technische Universität Berlin, Germany

Optimal Path Planning for soaring flight

Lucas Bonin Master Student, ENAC, French Civil Aviation University, Toulouse, France. bonin.lucas.74@gmail.com

Daniel Delahaye Professor, OPTIM Team Director, ENAC, French Civil Aviation University, Toulouse, France. delahaye@recherche.enac.fr

Andreas Guitart PhD Student, ENAC, French Civil Aviation University, Toulouse, France.andreas.guitart@enac.fr

Eric Feron Professor Kaust, Saudi-Arabia. eric.feron@kaust.edu.sa

Xavier Prats Dept. of Physics/Aeronautics Division,Technical University of Catalonia UPC/BarcelonaTECH Castelldefels, Spain. xavier.prats@upc.edu

ABSTRACT

In the last couple of years, performances of light soaring aircraft (para gliders, hang glider or light sailplane) have increased significantly, allowing pilots to fly great distances using only the convective energy of the atmosphere. This activity, called "Cross-Country flying" or "soaring", requires topological and aerological knowledge and a lot of pre-flight preparation in order to make the right decisions, and thus maximize the flying distance or minimize the flying time in a race. To optimize these flights, a pre-tactical decision support tool has been developed. This tool results from the adaptation of a sampling-based algorithm (FMT*). It is extended in such a way it can deal with differential constraints associated with light soaring aircraft, operating in a convective atmosphere, in a field of wind and close to reliefs. The method has been validated on real light soaring aircraft trajectories.

Keywords: Motion planning, trajectory Design, soaring aircraft, anisotropy

Nomenclature

Φ, γ	=	heading, glide path angle
$\chi, \chi_{free}, \chi_{goal}$	=	state space, free space, goal region

1 Introduction

For the last years, ultralight gliders have evolved significantly making them more efficient and accessible. Current performances allow pilots to fly large distances without engines, using only the convective energy of the atmosphere. This activity, called "Cross-Country (XC) Flying", is mainly based on flight trajectories that join several thermals (i.e., columns of air with significant upwind components). Spiral-like trajectories are flown inside thermals in order to take enough altitude and then glide towards the next thermal area, repeating this process in order to travel long distances (Figure 1).

Year after year, more and more XC competitions take place. All competitors are equipped with cutting-edge technology and it has become very challenging to win. At present, there are two ways to compete in XC :

- to fly the longest distance (Open distance).
- to fly the first to finish (Race to goal).

When competing in Race to goal, pilots are supposed to fly all the way through specified control points represented by virtual cylinders in the air. Both competitions require very specific flying skills, along with an accurate knowledge of the meteorological conditions.

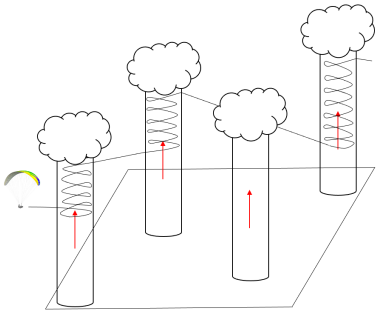


Fig. 1 Cross-Country (XC) flying

In order to give the pilot an idea of what could be the optimal path to follow for such a competition, a pre-tactical decision support tool is proposed in this paper. The tool is based on a modified fast marching tree (FMT*) algorithm. This sampling-based method takes into account the wind forecast of the competition day and a probability map of thermals [1]. However, the model presented does not take into account the flight dynamics of the ultralight glider and considers weather conditions as constant over time.

The main contribution of this paper is to model the path planning problem for XC flying as an optimal kinodynamic motion planning problem. The second contribution of this paper is to provide an extension of the FMT* algorithm to deal with high anisotropic environments and nonlinear differential motion constraints. Usually, the drawback of adding differential motion equations in sampling-based algorithm is the need of two-point boundary value problem (TPBV) solver that computes a near optimal control input to steer the vehicles between specified initial and final states. As way to get around, the TPBV has been reduced to either a quadratic or non-linear equation.

2 Previous Related Works

The main objective of motion planning problems is to find a sequence of valid configurations that drives an object (i.e., vehicle) from an origin to a destination point, while avoiding collision with obstacles and minimising a given cost function (time, distance,..). Several approaches and methods have been developed to address this problem. The most important ones are outlined below.

2.1 Discretization of the space

One approach is to discretize the research space in the form of graph or a tree and use this structure to find a feasible path.

2.1.1 Deterministic algorithms

The first category of discreet methods are deterministic algorithms, where the network is fixed and two same sets of initial condition generates the same solution. The solution is limited as it depends on the graph. A larger problem or a continuous research space can lead to a high number of node and thus, an important computational time. Dijkstra's algorithm, introduced by Edsger Dijkstra in 1959 [2], is a classical way to find the shortest path in a graph with positive edge values. For a given starting node, the algorithm will compute the minimum cost path to any node of the graph. Using a priority queue, the time complexity is $O(a+n.log(n))$ with a the number of edges and n the number of nodes. The A* Algorithm is an informed path search algorithm [3], often used when it comes to plan a robot path. It is close to the Dijkstra's method, where a heuristic function f is added and steer the path towards the goal:

$$f(n) = g(n) + h(n) \tag{1}$$

with g the cost of the path from the start and h, an estimation of the cost from the node n to the goal. At each iteration, the node n that minimizes f is added, in such a way it extends the path all the way to the goal. Since the heuristic function must be admissible in order to ensure its optimality, this is not an easy task for a complex problem.

2.1.2 Probabilistic algorithms

The other category of discreet algorithms are probabilistic one. Instead of using an explicit representation of the space, a set of points is randomly sampled in the free space. Then, a research tree is built to explore the space and to find a feasible trajectory. Compared to deterministic methods, these algorithms are less costly from a computational time point of view and more adapted to high dimensional space. The sampling-based methods is a family of methods that includes multi-query algorithms, such as *Probabilistic Road Map (PRM)*; and single query methods, such as *Rapidly Exploring random Tree (RRT)* [4]. It also includes their asymptotically optimal equivalents PRM* and RRT*. A state-of-the-art sampling-based algorithm is the fast marching tree FMT* [5]. It is asymptotically optimal and converge significantly faster than its counterpart path planning algorithms, justifying its use in the article.In the PRM a random graph is computed in order to represent the research space and then, a classical path searching method is applied to this graph. For RRT and FMT, the rationale is different: the sampling is used to build a research tree iteratively, from a start to the goal. An extension of the sampling-based algorithms RRT* and FMT* have been developed in [6] and in [7] to handle systems described by differential constraints.

2.2 Optimal control

Another approach to answer the motion planning problem is to solve the associated optimal control problem (OCP) where the goal is to compute, for any time, the optimal input for a system in order to minimize an objective function, while taking into account the dynamics of the system by considering, as constraints of the optimisation, a set of differential equations. Methods to solve OCP are based on either direct or indirect methods, among which, level-set methods, collocation, multiple shooting. One way to solve the OCP is to transcript it into a non linear problem and solve it using standard non linear programming (NLP) solvers [8]. Another way to solve OCP is to make the analogy with the propagation of a wave. Since wave tend to propagate along the path whose traveling time is shortest. By simulating the propagation of a wave, an optimal path can be determined, based on a criterion that has to be optimized.

Fast Marching The *Fast Marching* algorithm presented in [9] is part of more general method called level set algorithm. It determines the minimum time to reach any point of the space from a starting point. This method can be applied in case of isotropic environment or if the anisotropy is low enough.

Ordered Upwind When the propagation of the front depends on the position and on the direction of motion, the space is not considered as isotropic. It is the case under a wind field, if the wind speed has the same order of magnitude than the vehicle's speed. The *Ordered Upwind* algorithm has been developed in [10] to overcome this problem. However, it is more costly compared to the Fast Marching algorithm.

2.3 Trajectory optimization for soaring aircraft

One of the first research on the optimization of XC flying was made by Paul MacCready in 1954 [11]. He introduced the speed to fly as the optimal airspeed to adopt between two thermals making some assumptions on the next thermal vertical speed. Very popular for sailplane, this method is less common

in paragliding since pilots do not typically have an airspeed indicator. Also, this method does not allow an overall optimization of the path, since it considers each leg (from thermal to thermal) independently. Since the introduction of the MacCready speed to fly, several ways have been studied to optimize the trajectory of soaring vehicles in the atmosphere:

- Integrate the dynamic equation of glider from a starting point with a set of control. [12]
- Consider the Hamilton-Jacobi-Bellman equation for the optimal speed to fly [13]
- Solve the optimal control problem of minimizing the altitude loss of a glider manoeuvring in still air[14]

These above approaches deals with path planning and some of them makes assumptions on the wind field that's surround the aircraft. However, the possibility to optimize the trajectory without assuming the characteristics of the aerological environment have been studied in [15] or in [16]. The method used in [15] provides an autonomous soaring for a small glider using Q-learning, a Reinforcement Learning algorithm. However, since the paraglider dynamic is complex, the Q-learning algorithm is not adapted. In [16], the glider uses visual sensor to detect clouds and deduce navigation and guidance.

3 Problem Setup

This section formally formulates the problem to be solved and details the different models that are considered.

3.1 Atmospheric model

The air vehicle evolves in convective atmosphere and in a wind field which are supposed constant over time for a given 3D location. The two are set as one 3D vector field $W : \mathbb{R}^3 \to \mathbb{R}^3$, the updraft being modeled by the vertical wind speed:

$$(x,y,z) \mapsto (W_x(x,y,z), W_y(x,y,z), W_z(x,y)) \tag{2}$$

The longitudinal components W_x, W_y are computed using a weather forecast, while for the vertical component W_z, there are two possibilities. The first one is to use weather forecast with a high resolution mesh grid, like the one generated with WRF (Weather Research and Forecasting tool), which directly gives the vertical velocity. The other way is to use a thermal map [1], where the position of the center of thermals are indicated and noted $(x_i, y_i), \forall i \in N$. In order to compute W_z from this thermal map, each thermal is modeled as 2D Gaussian independent of the z coordinate and the maximum altitude of a thermal is limited by the cloud ceiling noted z_{max}. Thus, for N thermals the vertical component $\forall (x,y) \in \mathbb{R}^2$ is written as:

$$W_z(x,y) = \begin{cases} A \sum_{i=1}^{N} exp(-(\frac{(x-x_i)^2}{2\sigma^2} + \frac{(y-y_i)^2}{2\sigma^2})) & \text{if } z \leq z_{max} \\ 0 & \text{if } z > z_{max} \end{cases} \tag{3}$$

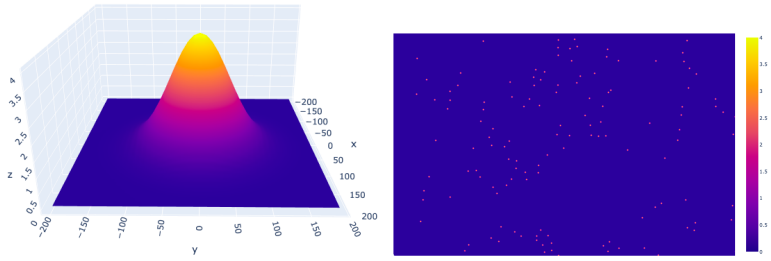
with σ the order of magnitude of the radius of a thermal and A the maximum velocity inside thermals (see Figure 2).

3.2 Terrain

The glider evolves over the ground and should avoid the collision with it. Thus, let us define the terrain as a smooth surface $z_t : \mathbb{R}^2 \to \mathbb{R}$ that is computed by interpolating real data points with a piecewise cubic, C1 smooth, curvature-minimizing interpolant in 2D (see Figure 3).

3.3 Soaring aircraft kinematics

During the flight, the pilot can adjust the following control variables (see figure 4 and 5):



(a) Gaussian model of thermal (b) Thermal map

Fig. 2 Modeling approach for thermals

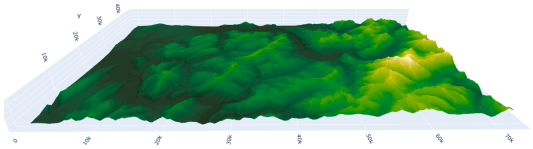


Fig. 3 Earth surface

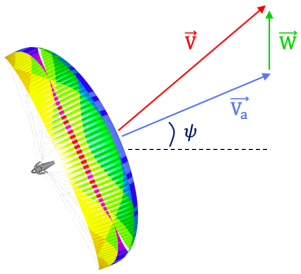


Fig. 4 Heading control

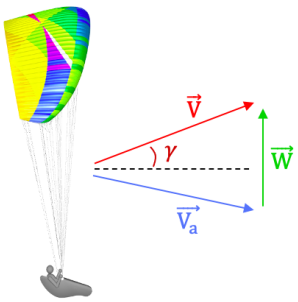


Fig. 5 Glide path angle control

- the heading $\psi \in [0, 2\pi]$;
- the airspeed $V_a \in \mathbb{R}^3$, in order to modify the glide path angle $\gamma \in [-\pi, \pi]$.

Let v_h and v_v be the horizontal airspeed and the vertical airspeed of the glider such that:

$$V_a = (v_h \cos\psi, v_h \sin\psi, v_v) \tag{4}$$

The airspeed, the ground speed and the wind field are linked through the following differential equations which describe the motion of the aircraft in the space:

$$\dot{x}(t) = v_h(t)\cos\psi(t) + W_x(x,y,z) \quad \dot{y}(t) = v_h(t)\sin\psi(t) + W_y(x,y,z) \quad \dot{z}(t) = v_z(t) + W_z(x,y) \tag{5}$$

with $(x(t), y(t), z(t))$ the position at t of the aircraft in a Cartesian frame. The real kinematic behaviour of light gliders is complex and depends on its nature (para glider, hang glider or sail plane). The model described here is simplified and assumptions are made:

Assumption 1. *Apparent mass or inertia effect are neglected.* This means that the vehicle is always in steady-state condition.

Assumption 2. *The airspeed of the glider is described by its speed polar.* This means that at any time, the vertical speed v_v and the horizontal speed v_h are linked by the polar curve (Figure 6). This mean that $v_v = p(v_h)$, where the function p can be considered as a quadratic function ($\forall x \in \mathbb{R}^+, p(x) = k_2x^2 + k_1x + k_0$) or a linear one ($\forall x \in \mathbb{R}^+, p(x) = l_1x + l_0$) and $v_{h_{min}} \leq v_h \leq v_{h_{max}}$. In still air, the glider is always in descent (the flight path angle is always negative). The local wind field due to a thermal will affect the ground glide path angle, allowing the glider to gain altitude in the case of sufficient upwind.

Assumptions 3. *The speed polar is independent from the turn rate (may be true for shallow turn).*

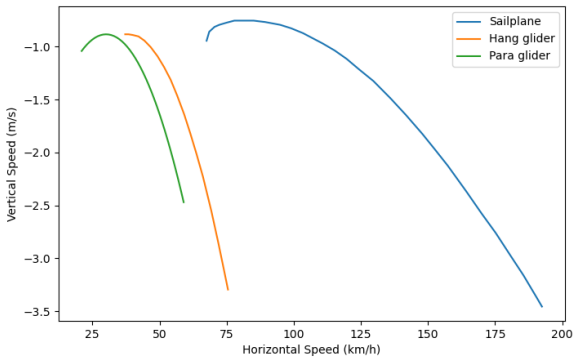


Fig. 6 Speed polar of soaring aircrafts

3.4 Optimal motion planning problem

Finding the minimum time trajectory between two points in XC competitions can be expressed as an optimization problem as follows:

- **Optimality Criterion:** Minimize the flying time from the starting point (x_0, y_0, z_0) to the goal point (x_f, y_f, z_f);
- **State Variables:** Position of the aircraft at the time instant t : $(x(t), y(t), z(t))$;
- **Control Variables :** At time t, the horizontal speed $v_h(t)$ and the heading $\psi(t)$. $\forall t > 0, v_h(t) \in [v_{h_{min}}, v_{h_{max}}], \psi(t) \in [0, 2\pi]$;
- **Constraints:** Respect the differential equations (5), the speed polar and avoid collision with the terrain (i.e., the height of the glider is always strictly positive).

Mathematically, this problem can be written as:

$$\begin{aligned}
&\min_{\psi(.), v_h(.)} \quad \int_0^T \mathrm{d}t = T \\
&\text{subject to:} \quad \dot{x}(t) = v_h(t)\cos\psi(t) + W_x(x(t), y(t), z(t)) \\
&\qquad\qquad\quad \dot{y}(t) = v_h(t)\sin\psi(t) + W_y(x(t), y(t), z(t)) \\
&\qquad\qquad\quad \dot{z}(t) = p(v_h(t)) + W_z(x(t), y(t)) \\
&\qquad\qquad\quad v_{h_{min}} \leq v_h(t) \leq v_{h_{max}} \\
&\qquad\qquad\quad z_t(x(t), y(t)) < z(t) \\
&\qquad\qquad\quad (x(t_0), y(t_0), z(t_0)) = (x_0, y_0, z_0) \\
&\qquad\qquad\quad (x(t_f), y(t_f), z(t_f)) = (x_f, y_f, z_f)
\end{aligned} \tag{6}$$

This optimization problem can be expressed in the general form of an optimal kinodynamic motion planning problem as the one presented in [6]. Let the state space $\chi \subset \mathbb{R}^3$ and the control space $U \subset \mathbb{R}^2$ be compact sets such that :

$$\forall X(t) \in \chi \quad X(t) = (x(t), y(t), z(t)) \quad \text{and} \quad \forall u(t) \in U \quad u(t) = (v_h(t), \psi(t)) \tag{7}$$

Let the subset $\chi_{obs} \subset \chi$ be the *obstacle region* such that χ / χ_{obs} is an open set and $\chi_{free} = \chi / \chi_{obs}$ define the *obstacle-free region* such that:

$$\forall X(t) \in \chi_{free} \iff z_t(x(t), y(t)) < z(t) \tag{8}$$

Let $\chi_{goal} \subset \chi$ be the *goal region* and the cost functional :

$$\begin{aligned}
g \; : \; \chi \quad &\to \quad \mathbb{R}^+ \\
X(t) \quad &\mapsto \quad g(X(t)) = 1
\end{aligned} \tag{9}$$

Finally, let $\mathbb{X}$ and $\mathbb{U}$ be the sets of all essentially bounded measurable functions defined $\forall T \in \mathbb{R}^{+*}$ from $[0,T]$ to χ and U, respectively. Then, the problem writes as follows [6]: *Given the state space χ, the goal region χ_{goal}, the obstacle-free region χ_{obs}, an initial state $(x_0, y_0, z_0) \in \chi_{free}$ and a differential equation that describe the system dynamics. Find a control $u \in \mathbb{U}$ with domain $[0,T]$ for $T \in \mathbb{R}^{+*}$ such that the trajectory $X \in \mathbb{X}$:*

- verifies $\dot{X} = f(X(t), u(t)), \forall t \in [0,T]$;
- avoid obstacles, $X(t) \in \chi_{free}, \forall t \in [0,T]$;
- reaches the goal region, $X(T) \in \chi_{goal}$;
- minimize the cost function $J(X) = \int_0^T g(X(t))\mathrm{d}t$.

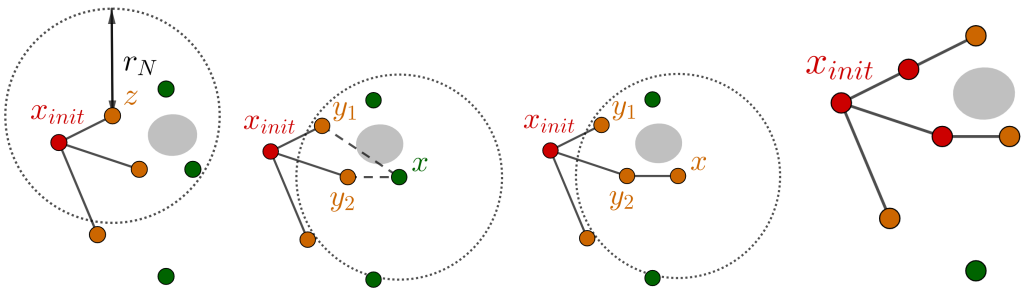
4 Problem Resolution

The algorithm used to solve the problem presented above is an adaptation of the sampling-based algorithm *FMT** [5]. Let us outline the main procedure that the algorithm relies on.

4.1 FMT* Algorithm

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



(a) Step1 (b) Step2 (c) Step3 (d) Step4

Fig. 7 One iteration of the FMT* algorithm.

Let us detail one iteration:

1) 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 (see figure 7(a)).
2) 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;see figure 7(a)).
3) 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} (see figure 7(c)).
4) Once all z unvisited neighbors have been visited, z is put in V_{closed} and the iteration is done (see figure 7(d)).

The FMT* (for which the pseudocode is given in Appendix) is extended in order to take into account the high anisotropic space. In the classical FMT*, new tree nodes are added in set named $V_{open,new}$ and this set is added to a global set V_{open} afterward. These steps don't allowed radical changes in the direction of the propagation. A way to get around is to add new tree nodes directly in the V_{open} set.

4.2 Implementation details

4.2.1 Non Uniform Sampling Strategy

During a flight, the behavior of the glider changes. When it comes to glide between thermals, the flight regime is constant and the trajectory is composed by long segments. On the other hand, during the climb, the aircraft makes narrow spirals. To account for this, thermal areas are sampled with more density than the free space as illustrated in figure 8.

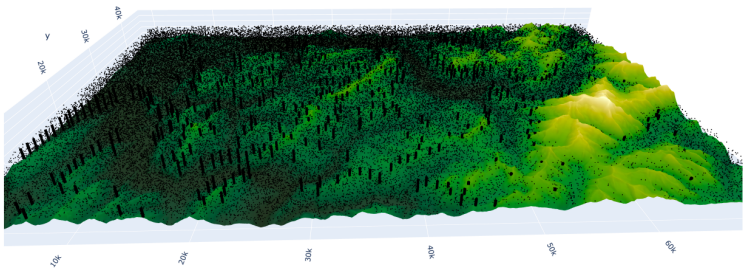


Fig. 8 Non-uniform sampling of the 3D space

To ensure asymptotic optimality, the neighbour radius r must respect the following equation [5]:

$$r_n = 2(1+\eta)\left(\frac{1}{d.l}\right)^{\frac{1}{d}}\left(\frac{\mu(\chi_{free})}{\zeta_d}\right)^{\frac{1}{d}}\left(\frac{log(n)}{n}\right)^{\frac{1}{d}} \tag{10}$$

where:

- l is the strictly positive upper bound of the probability density function supported over χ_{free}.
- $\eta > 0$ is a tunning parameter which depends on the nature of the problem. In this case, the kinematic constraint is binding, the covered neighbours are few compare to the classical case. Thus, η must compensate this added constraint.
- μ is the Lebesgues measure and ζ_d is the volume of the unit ball in d-dimension.

4.2.2 Trajectory and Cost computation

In the FMT* algorithm, for two samples $(Z_1, Z_2) \in \chi_{free} \times \chi_{free}$ the function $Cost(Z_1, Z_2)$ return the cost of the optimal trajectory that starts from Z_1 and reaches Z_2, when such a trajectory exists. Compute such optimal trajectory, for a line integral cost, for every connection, can be computationally challenging. Indeed, it means to solve a two-point boundary value problem for each connection. Thus, to compute the cost between two states Z_1 and Z_2, the optimal path is approximated with a cost-weighted, **straight line connection** which respect the differential constraint. Along the straight line $\overline{Z_1 Z_2}$, the control is supposed constant and thus the solution for the control u is piecewise-constant.

$$Z_2 = Z_1 + \int_{t_1}^{t_2} \dot{X}(t).dt \qquad Z_2 = Z_1 + \int_{t_1}^{t_2} f(X(t), \overline{u}).dt \tag{11}$$

The integral is computed using a repeated midpoint rule. The interval $[t_1, t_2]$ is partitioned into m equally spaced points and $h = t_2 - t_1$

$$Z_2 = Z_1 + \frac{h}{m} \sum_{i=1}^{m} f(Z_1 + \frac{i(Z_2 - Z_1)}{m+1}, \overline{u}) \tag{12}$$

The straight line connection $\overline{Z_1 Z_2}$ is feasible with respect to the differential equation if and only if $\exists\, h > 0$ and $\exists\, \overline{u} \in U$ verifying the equation (12). Thus, compute the cost h is equivalent to solve the following system of equations :

$$\begin{aligned} x_2 &= x_1 + h\overline{v_h}\cos\overline{\psi} + \frac{h}{m}\sum_{i=1}^{m} W_x(x_{mi}, y_{mi}, z_{mi}) \\ y_2 &= y_1 + h\overline{v_h}\sin\overline{\psi} + \frac{h}{m}\sum_{i=1}^{m} W_y(x_{mi}, y_{mi}, z_{mi}) \\ z_2 &= z_1 + hp(\overline{v_h}) + \frac{h}{m}\sum_{i=1}^{m} W_z(x_{mi}, y_{mi}, z_{mi}) \end{aligned} \tag{13}$$

with p the *speed polar function* and $\forall i \in [1, m]$:

$$x_{mi} = x_1 + \tfrac{i(x_2 - x_1)}{m+1} \quad y_{mi} = y_1 + \tfrac{i(y_2 - y_1)}{m+1} \quad z_{mi} = z_1 + \tfrac{i(z_2 - z_1)}{m+1} \tag{14}$$

For the sake of simplicity the system (13) is written :

$$\mathrm{d}x = h(\overline{v_h}\cos\overline{\psi} + w_{xm}) \quad \mathrm{d}y = h(\overline{v_h}\sin\overline{\psi} + w_{ym}) \quad \mathrm{d}z = h(p(\overline{v_h}) + w_{zm}) \tag{15}$$

4.2.3 Reachable set

Exact estimation of the reachable set is also a computationally intensive task. However, the reachable set can be estimate. For a set V and a state Z, $Reachable(V, Z)$ returns the subset $\Gamma \subset V$ such that $\forall Z' \in \Gamma, \exists h > 0, \exists \overline{v_h} \in [v_{h_{min}}, v_{h_{max}}], \exists \overline{\psi} \in [0, 2\pi]$ solution of (15).

4.2.4 Cost functional with linear polar

In the case of linear speed polar $p(x) = l_1 x + l_0$, solve (15) is equivalent to solve the quadratic equation :

$$ah^2 + bh + c = 0 \tag{16}$$

with

$$\begin{aligned} a &= l_1^2({w_{xm}}^2 + {w_{ym}}^2) - (l_0 + w_{zm})^2 \\ b &= 2(\mathrm{d}z(l_0 + w_{zm}) - l_1^2(\mathrm{d}x w_{xm} + \mathrm{d}y w_{ym})) \\ c &= l_1^2(\mathrm{d}x^2 + \mathrm{d}y^2) - \mathrm{d}z^2 \end{aligned} \tag{17}$$

Thus, for two states Z_1, Z_2, $Cost(Z_1, Z_2)$ return the solution $h > 0$ of (16) which verifies:

$$\overline{v_h} = \frac{1}{l_1}(\frac{dz}{h} - (l_0 + w_{zm})) \in [v_{h_{min}}, v_{h_{max}}]$$

4.2.5 Cost functional with quadratic polar

In the case of quadratic speed polar $p(x) = k_2.x^2 + k_1.x + k_0$, the system (15) can be reduce to the following equation :

$$(\frac{\mathrm{d}x}{\mathrm{d}z}(p(\overline{v_h}) + w_{zm}) - w_{xm})^2 + (\frac{\mathrm{d}y}{\mathrm{d}z}(p(\overline{v_h}) + w_{zm}) - w_{ym})^2 = \overline{v_h}^2 \tag{18}$$

which can solve using bisection method. Thus, for two states Z_1, Z_2, $Cost(Z_1, Z_2)$ return the solution $\overline{v_h} \in [v_{h_{min}}, v_{h_{max}}]$ of (18) which gives the minimum $h > 0$.

5 Results and Discussion

The algorithm presented above was implemented using Python and ran on a computer with a 1,8 GHz processor, 8 Gb of RAM and a MacOS operating system. For the following experiments, the linear polar function representing a competition paraglider was chosen :

$$v_v = -0.2104 v_h + 1.0247 \quad v_{h_{max}} = 16 m/s \quad v_{h_{min}} = 8 m/s \tag{19}$$

Since the optimal solution of the real 3D problem is not known the algorithm is firstly validated on 2D scenarios, both on the longitudinal and the vertical axis. Then a real problem is solved and the results of the algorithm are compared with the execution of a real flight.

5.1 Longitudinal Validation : Zermelo's Navigation problem

The following problem is to plan a 2D trajectory in the horizontal plane from a starting point $A = (20000, -5000)$ to a final point $B = (0,0)$ while minimising the travelling time. In order to allow the glider to move horizontally, the upwind $W_z(x,y) = \overline{w_z}$ is set constant equal to 1.5 m/s. As $\mathrm{d}z = 0$, v_h is constant :

$$\overline{v_h} = -\frac{l_0 + w_{zm}}{l_1} = 12\ m/s$$

The horizontal wind is set as following :

$$W_x(x,y) = W_x(y) = -\frac{\overline{v_h} y}{h} \qquad W_y(x,y) = 0 \tag{20}$$

with $h = 5000m$. This problem is known as the Zermelo's navigation problem and in this particular case ($W_x(x,y) = f(y), W_y(x,y) = 0$), it admits a closed form (analytical) solution. The solution is obtained by using the Pontryagin's maximum principle. The optimum time solution for this problem is **1751.289 s**. The algorithm 1 presented in section 4 was ran with different uniform samplings and results were compared with the optimal (analytical) trajectory (see figure 9). As the number of samples increases, the computed trajectory with FMT* converges toward the optimum solution (see figure 10).

5.2 Vertical Validation

Typical trajectories in the vertical plane in XC flying are represented in figure 11. The trajectory consists of a series of gliding phases and climbing phases within thermal areas. The structure of the minimum time trajectory depends highly on the position of thermal areas and their intensity. If thermals are weak, the pilot should fly at low speed to decrease the sink rate and thus decrease the altitude loss (blue trajectory). On the contrary, with powerful thermals with high upwind speed, the pilot should fly at high speed because (s)he will recover the loss of altitude quickly in the next updraft (black trajectory). To verify that the algorithm approximates well the optimal trajectory on the vertical plane, the following scenario has been put in place. The problem is to plan the minimum time trajectory from a point $A =$

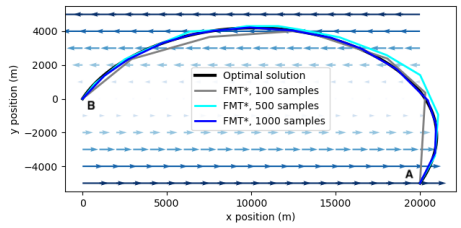


Fig. 9 Comparison of the computed trajectories with FMT* and the optimal trajectory, from A to B.

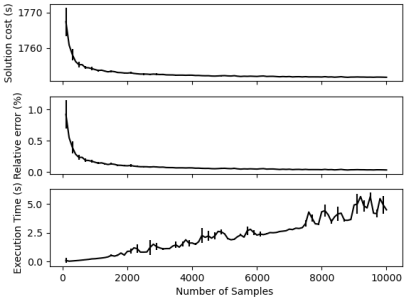


Fig. 10 Evolution of the solution with the number of samples.

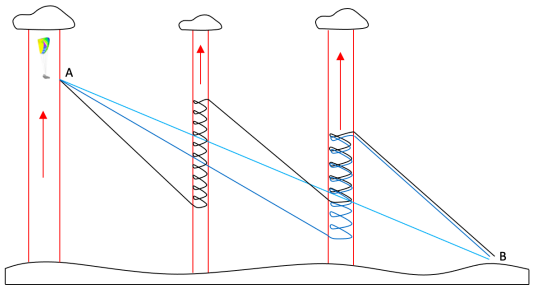


Fig. 11 Different vertical trajectories in XC flying from A to B.

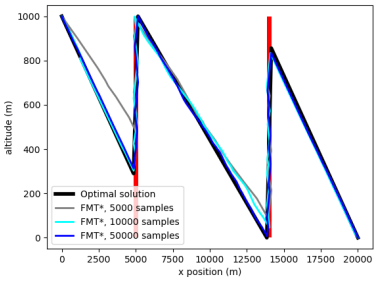


Fig. 12 Vertical profiles comparison.

$(0, 1000)$ to a point $B = (20000, 0)$ while going through different cylindrical thermal areas with a radius equal to 150m. The longitudinal wind is null. The position of two thermals are [5000 m, 14000 m] with an upwind speed equal to 6 m/s. In order to access to the optimal solution, the problem can be expressed as non linear programming problem (NLP) where the decision variables are the altitude entrance and exit of each thermals and the speed v_h for each gliding segments. The speed v_h in thermal areas is constant and equal to the one that gives the minimum sink rate. Then, the optimal cost and trajectory are computed using a nlp solver. In this example, the optimal time is **1772.39 s**. To point out the importance of the non uniform sampling in anisotropic environment, the FMT* algorithm was run with different samplings. Thermal areas, noted $\chi_{thermal} \subset \chi_{free}$, are sampled with m samples and χ_{free} is sampled with n samples. Let us denote $\alpha = \frac{m}{n}$ the anisotropic factor of the sampling. Figure 12 compares the computed trajectories with the optimal one. With $\alpha = 0$, the sampling is uniform. In this case, a high number of samples and thus a high computational time is necessary to approach the optimal solution. With $\alpha > 0$, the sampling is non uniform and the convergence much faster. As pointed out above, the relative gap between the computed time and the optimal time is decreasing when the number of samples increases (see figure 13). Due to the fact that the environment is highly anistropic, the number of samples has to

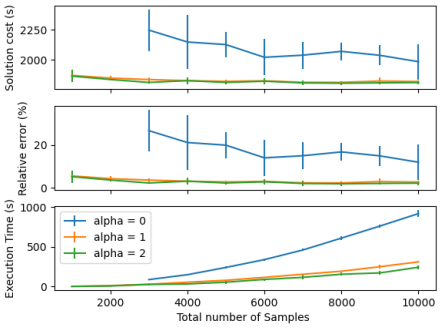


Fig. 13 Evolution of the solution with the number of samples.

be high to approximate well the solution. The previous two sections showed that the proposed method gives correct results on both axis, close to the optimal solution, if the algorithm is set with a sufficient number of samples.

5.3 Comparison of real and computed trajectories

The following section deals with the real 3D scenario over the ground surface. Three different real trajectories (available at https://www.xcontest.org/world/en/) were compared to computed ones. The simulated trajectories show important gains in terms of flight time compared to the real ones (Table 1). However, computed trajectories rely on a theoretical thermal map and on wind forecast which can be far from reality. Examples of trajectories produced for the first case are shown in figures 14 and 15.

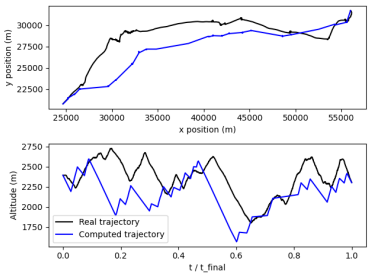


Fig. 14 Comparison of the longitudinal trajectories and the altitude profiles for flight n°1.

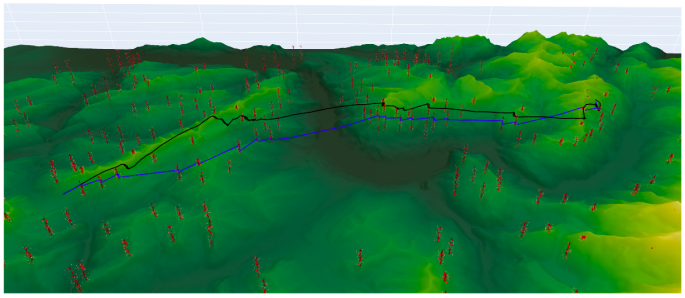


Fig. 15 3D plots of the computed trajectory and the real trajectory for flight n°1, thermal areas are represented by red quivers.

Table 1 Cost comparison of computed trajectories with FMT* and real trajectories

Id	Computed Solution (s)	Real trajectory (s)	Gain (%)
1	3345.28	4919	32
2	3524.3	5392	34.6
3	3265.7	3565.7	8.4

6 Conclusion

In this paper, a new model of path planning for soaring aircraft has been presented as well as an anisotropic extension for the FMT* algorithm. Simulations point out that the method is well adapted to plan the path in a 3D wind field with complex kinematic of motion while avoiding terrain collision. However, several points have to be improved. Because the quality of the solution depends highly on the accuracy of the probabilistic weather data (the wind and thermal map), a confidence index associated with each computed trajectory could be provided. Moreover, stochastic or robust approaches, as proposed in [17], could be an interesting approach to further explore. Also, the computational time for the 3D case is the order of few hours. Swiftness could be improved by the use of *C* or *JAVA* instead of *Python* and a better computer configuration. Any aircraft must respect the rules of the air and thus, avoid restricted airspace. These areas have to be added in the obstacle set. Finally, the weather (wind and thermals) depends on time. For trajectories longer than 2 hours, the time variability should also be taken into account.

Appendix

Pseudo code of the FMT* algorithm.

Algorithm 1 FMT* with anisotropic extension

$V \leftarrow \{x_{init}\} \cup SampleFree(n)$; $E \leftarrow \emptyset$
$V_{unvisited} \leftarrow V \setminus \{x_{init}\}$;$V_{open} \leftarrow \{x_{init}\}$;$V_{closed} \leftarrow \emptyset$
$z \leftarrow x_{init}$
while $z \notin \chi_{goal}$ **do**
    $N_z \leftarrow Near(V \setminus \{z\}, z, r)$
    $X_{near} \leftarrow N_z \cap V_{unvisited}$
    for $x \in X_{near}$ **do**
        $N_x \leftarrow Near(V \setminus \{x\}, x, r)$
        $\Gamma_x \leftarrow Reachable(V \setminus \{x\}, x)$
        $Y_{near} \leftarrow N_x \cap V_{open} \cap \Gamma_x$
        if $Y_{near} \neq \emptyset$ **then**
            $y_{min} \leftarrow argmin_{y \in Y_{near}} \{c(y) + Cost(y,x)\}$
            if $CollisionFree(y_{min}, x)$ **then**
                $E \leftarrow E \cup \{(y_{min}, x)\}$
                $V_{open} \leftarrow V_{open} \cup \{x\}$
                $V_{unvisited} \leftarrow V_{unvisited} \setminus \{x\}$
                $c(x) \leftarrow c(y_{min}) + Cost(y_{min}, x)$
            end if
        end if
    end for
    $V_{open} \leftarrow V_{open} \setminus \{z\}$
    $V_{closed} \leftarrow V_{closed} \cup \{z\}$
    $z \leftarrow argmin_{y \in V_{open}} \{c(y)\}$
end while
return $Path(z, T = (V_{open} \cup V_{closed}, E))$

References

[1] Michael von Kanel. Paraglidingnet: A sensor network for thermal research. Master's thesis, Swiss Federal Institute of Technology Zurich, vkaenemi@ee.ethz.ch, 8 2010.

[2] Edsger W Dijkstra et al. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[3] Peter E Hart, Nils J Nilsson, and Bertram Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[4] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7):846–894, 2011.

[5] Lucas Janson, Edward Schmerling, Ashley Clark, and Marco Pavone. Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions. *The International journal of robotics research*, 34(7):883–921, 2015.

[6] Sertac Karaman and Emilio Frazzoli. Optimal kinodynamic motion planning using incremental sampling-based methods. In *49th IEEE conference on decision and control (CDC)*, pages 7681–7687. IEEE, 2010.

[7] Edward Schmerling, Lucas Janson, and Marco Pavone. Optimal sampling-based motion planning under differential constraints: the driftless case. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2368–2375. IEEE, 2015.

[8] John T Betts. *Practical methods for optimal control and estimation using nonlinear programming*. SIAM, 2010.

[9] James A Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.

[10] James A Sethian and Alexander Vladimirsky. Ordered upwind methods for static hamilton–jacobi equations. *Proceedings of the National Academy of Sciences*, 98(20):11069–11074, 2001.

[11] Paul B. McCready. An optimal airspeed selector. *Soaring*, 1954.

[12] Nicholas RJ Lawrance. Autonomous soaring flight for unmanned aerial vehicles. 2011.

[13] Robert Almgren and Agnès Tourin. Optimal soaring via hamilton–jacobi–bellman equations. *Optimal Control Applications and Methods*, 36(4):475–495, 2015.

[14] Artur Wolek and Craig Woolsey. Optimal paths in still air for a sailplane with a quadratic glide polar. *Technical Soaring*, 40(2):9–23.

[15] Erwan Lecarpentier, Sebastian Rapp, Marc Melo, and Emmanuel Rachelson. Empirical evaluation of a q-learning algorithm for model-free autonomous soaring. *arXiv preprint arXiv:1707.05668*, 2017.

[16] Martin Stolle. *Towards Vision-Based Autonomous Cross-Country Soaring for UAVs*. PhD thesis, UNIVERSITE DE TOULOUSE, 2017.

[17] Daniel González-Arribas, Manuel Soler, and Manuel Sanjurjo-Rivo. Robust aircraft trajectory planning under wind uncertainty using optimal control. *Journal of Guidance, Control, and Dynamics*, 41(3):673–688, 2018.