IEEE Control Systems Letters paper presented at
2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

General extremal field method for time-optimal trajectory planning in flow fields

Bastien Schnitzler[1], Antoine Drouin[2], Jean-Marc Moschetta[3] and Daniel Delahaye[4]

***Abstract*— We present an algorithm to compute time-optimal trajectories for light vehicles in unsteady flow fields, with applications to long-range, low-power aircraft as well as underwater vehicles in ocean currents. The proposed approach aims at unifying various works from the literature on extremal fields and extends it by several features. First, we propose an exact scheme to deal with still obstacles. While being directly useful for pure obstacles, it is also of particular interest to ensure the validity of computation at the borders of the problem domain. Second, we demonstrate the method ability to deal with trajectory planning for long-range airborne missions with real weather data. Lastly, the source code, written in Python, is made open to the community to accelerate research in the domain.**

I. Introduction

The need to reduce CO_2 emissions in the aerospace industry led to the creation of the "Mermoz challenge", starting at ISAE-SUPAERO. This challenge is about designing an Unmanned Aerial Vehicle (UAV) to cross the Atlantic following air mail pioneer Jean Mermoz's route from Dakar, Senegal to Natal, Brazil, with limited carbon emissions. So far, the fuel cell solution based on liquid hydrogen was selected as power source for this $3000\,\mathrm{km}$ flight. By design, the UAV is light and its airspeed is low compared to commercial aviation aircraft crossing the Atlantic over similar routes. Thus, it is sensible to its environment: wind fields and convective weather areas. While the latter can be seen as a threat to the fulfillment of the mission, there is nevertheless an opportunity to save energy by a careful trajectory planning in the wind fields. This optimization process carries a lot of challenging features: spatially non-uniform and unsteady flow field, presence of hazardous zones, uncertainty in the weather data used for computation over a large time window. These features as well as the underlying model (specified in Section III) are shared with trajectory planning problems for underwater vehicles, so the literature is indeed built upon both application fields.

II. Literature review

Trajectory optimization for vehicles in flows has a rich literature arising from the diversity of optimization methods. A fundamental division appears: either the problem is discretized first, and optimized then, in which case we talk about *direct* methods, or the problem is first optimized theoretically and then cast to discrete representation, in which case we talk about *indirect* methods.

a) Direct methods: The most common direct method family is *control parameterization*. By discretizing state and control, it turns the trajectory planning problem into a Non-Linear Program (NLP) for which powerful solvers exist. This formulation is very versatile and has been used successfully even for large problems [5]. However, it finds *local* optima by design. *Graph-based* cost minimization methods are also popular in the literature. The popular robotic algorithm A* proved efficient for steady-flow cases [7]. Fast Marching (FM) methods also tackle the path planning problem successfully. They were applied to underwater vehicles [10] and their refinement in flow fields called *ordered upwinds* proved efficient for aircraft trajectory optimization [4]. Nevertheless, the controllability condition, *i.e.* the vehicle speed being greater than the flow field magnitude is often necessary for these methods to work.

b) Sampling-based methods: At the interface between direct and indirect methods, sampling-based methods carry out simultaneously a random sampling of the state space and optimization. They originated with probabilistic roadmaps, but really thrived with Rapidly-exploring Random Trees (RRT, RRT*) [11] as well as a sampling-based FM method called Fast Marching Trees (FMT*) [6]. For UAV guidance in 3D complex wind a "kinematic tree" method also proved efficient [2].

c) Indirect methods: This family of methods not only provide schemes for optimal trajectory computation but also enable a detailed theoretical look on what optimality means for the navigation problem. Mathematically, two approaches exist to find an optimal trajectory: either *characterize* it (sufficient conditions) or *eliminate* non-optimal trajectories (necessary conditions). We see this fundamental difference occur in the literature: the first approach entails Level-Set Methods (LSM) while the second leads to Extremal Field Methods (EFM). LSM have proved very efficient to deal with strongly unsteady flows [8]. They are also able to deal with uncertainty in the flow data [13]. On the other hand, applying necessary conditions let one deal with *extremal trajectories*. These trajectories are valid candidates to optimality among which the true optimum shall be sorted out [1]. An efficient scheme to do so is the construction of an *extremal field*, which was addressed in [12] and which is the focus of this article. This scheme was also cast to spherical geometry in [9].

III. Problem definition

Notations

In what follows, the norm of vector $\mathbf{x}$ is noted $|\mathbf{x}|$, the transpose of matrix A is noted $A^\intercal$. The set of continuous functions (resp. piece-wise continuous functions) defined

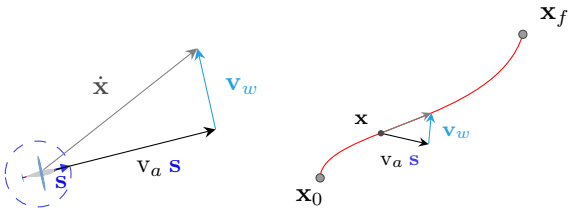

Fig. 1. Left, the basic kinematics of the situation linking the airspeed v_a, the heading vector $\mathbf{s}$ and wind speed $\mathbf{v}_w$ to ground speed $\dot{\mathbf{x}}$. Right, the control problem where the vehicle starts at $\mathbf{x}_0$ and has to reach $\mathbf{x}_f$ using a control law in airspeed $\mathrm{v}_a(\cdot)$ and in heading $\mathbf{s}(\cdot)$ minimizing some criteria.

on $[a, b]$ which may be scalar- or vector-valued is noted $\mathcal{C}^0([a, b])$ (resp. $\mathcal{C}^0_{\text{pw}}([a, b])$).

A. Model

We first formalize the trajectory planning problem features as they are seen from a large-scale point of view. We indeed focus on trajectory planning for the whole flight as opposed to local vehicle steering (*e.g.* wind gradient harnessing or gust avoidance).

State space: Either a plane ($\mathbb{R}^2$) or a sphere ($\mathbb{S}^2$ associated to Earth)
Vehicle: A point in the *state space* with a given speed but no inertia.
Flow field: A vector field over the *state space* (tangent to the manifold in the spherical case).
Obstacles: Subsets of the *state space* that are not safe for the vehicle and that should be avoided.

The state space is always restricted to two degrees of freedom. This is motivated by several reasons: for large-scale cruise flights, aircraft usually choose an adapted cruising altitude and maintain it throughout the whole flight. In regards to underwater vehicles, the vertical profile of the trajectory is often imposed before planning the trajectory, so only bidimensional planning is required. This is also an operational constraint of the Mermoz challenge.

For the sake of clarity, we will now state the kinematic model in its planar version and refer the reader to [9] for spherical equations. We will also extensively use the vocabulary of aircraft but it shall always be remembered that every notion can be cast to underwater navigation by replacing "wind" by "ocean current".

The considered domain is a bounded subset $\mathcal{D} \subset \mathbb{R}^2$. In the reference frame, $\mathbf{x}(t) \in \mathcal{D}$ denotes the vehicle position at time t, $\dot{\mathbf{x}} := \frac{\mathrm{dx}}{\mathrm{d}t}$ its speed, $\mathbf{s} \in \mathbb{S}^1$ is the heading direction and v_a its speed relative to the flow (called "airspeed" in what follows). The flow field is noted $\mathbf{v}_w$ and is a function of time and space. The situation is summarized in Fig. 1 and leads to the following kinematics:

$$\dot{\mathbf{x}}(t) = \mathrm{v}_a(t)\,\mathbf{s}(t) + \mathbf{v}_w(t, \mathbf{x}(t)) \tag{1}$$

We will study the problem of starting from a given point $\mathbf{x}_0 \in \mathcal{D}$ and reaching $\mathbf{x}_t \in \mathcal{D}$ minimizing travel time. This problem is best known as *Zermelo's problem* in the literature. We assume the problem data is defined over a sufficiently large time window $[0, T^\circ]$. With T fixed in $[0, T^\circ]$, we define a *flight* as a solution to the Cauchy problem:

$$\begin{cases} \mathbf{x}(0) = \mathbf{x}_0 \\ \mathbf{x}(\cdot) \text{ satisfies } (1) \\ \text{with } \mathrm{v}_a(\cdot),\ \mathbf{s}(\cdot) \text{ in } \mathcal{C}^0_{\text{pw}}([0, T]) \end{cases} \tag{2}$$

From then it will be useful to define *flyability*. In brief, it characterizes whether a path drawn on the ground can be followed in the air taking into account the wind advection.

Flyability: We say that a ground path $\xi(\cdot) \in \mathcal{C}^0([0, 1])$, $\xi : [0, 1] \to \mathcal{D}$ is *flyable* if there exist a time window upper bound T, a flight $\mathbf{x}(\cdot)$ on $[0, T]$, and a *time-warping* function $\gamma : [0, T] \to [0, 1]$ such that

$$\forall t \in [0, T],\ \mathbf{x}(t) = \xi(\gamma(t))$$

A time-warping function $\gamma : [a, b] \to [a', b']$ satisfies i) $\gamma(a) = a'$, $\gamma(b) = b'$ ii) γ invertible iii) γ and γ^{-1} continuous. It is used to eliminate time parameterization from the definition of the path.

With this notion, we can *eliminate* the airspeed as control variable for time-optimal problems. Indeed, for physical reasons the airspeed is always bounded by some constant $\mathrm{v}_a^{(\max)}$. If a flight uses an airspeed law that is not saturated to the maximum value, *i.e.* $\mathrm{v}_a(\cdot) < \mathrm{v}_a^{(\max)}$ on some open interval, then we can prove that this flight, seen as a ground path, is still flyable with $\tilde{\mathrm{v}}_a(\cdot) = \mathrm{v}_a^{(max)}$. As a result, the new travel time is reduced, and the former trajectory is suboptimal compared to the new one. The demonstration relies on the fact that it is always possible to increase the airspeed on a flight while maintaining the same ground route by adjusting the heading.

We can then formalize the Time-optimal Navigation Problem:

$$\text{(TNP)} \qquad \begin{cases} \inf\limits_{\mathbf{s}(\cdot) \in \mathcal{C}^0_{\text{pw}}([0, T^\circ])} T \\ \dot{\mathbf{x}}(t) = \mathrm{v}_a\,\mathbf{s}(t) + \mathbf{v}_w(t, \mathbf{x}(t)) \\ \mathbf{x}(0) = \mathbf{x}_0 \\ \mathbf{x}(T) = \mathbf{x}_t \end{cases} \tag{3}$$

Note that solutions of this problem are heading control laws. We now have formulated an optimal control problem, for which variational calculus will provide a resolution scheme.

B. Optimality conditions

We assume the wind field is $\mathcal{C}^0$ in time and $\mathcal{C}^1$ in space. We look for absolutely continuous solutions of the TNP. We use Pontryagin's Maximum Principle (PMP) to derive necessary conditions of optimality [1]. We introduce an adjoint state $t \mapsto \mathbf{p}(t) \in \mathbb{R}^2$ and a parameter $\lambda \in \mathbb{R}$, and define the Hamiltonian of the system:

$$\begin{aligned} H : \ & [0, T^\circ] \times \mathbb{R}^2 \times \mathbb{R}^2 \times \mathbb{R} \times \mathbb{S}^1 \to \mathbb{R} \\ & (t, \mathbf{x}, \mathbf{p}, \lambda, \mathbf{s}) \mapsto \mathrm{v}_a\,\mathbf{p}^\intercal\,\mathbf{s} + \mathbf{p}^\intercal\,\mathbf{v}_w(t, \mathbf{x}) + \lambda \end{aligned} \tag{4}$$

We assume there exists at least one solution to the TNP and note T^* the optimal travel time. The PMP states that

if $\mathbf{s}^*(\cdot)$ is a solution of the TNP and $(\mathbf{x}^*(\cdot), \mathbf{p}^*(\cdot), \lambda^*)$ the associated optimal triplet (absolutely continuous), then:

i) *Point-wise minimization* : For a.e. $t \in [0, T^*]$,
$$\mathbf{s}^*(t) \in \arg\min_{\mathbf{s}\in\mathbb{S}^1} H(t, \mathbf{x}^*(t), \mathbf{p}^*(t), \lambda^*, \mathbf{s})$$

ii) *Adjoint state evolution* : For a.e. $t \in [0, T^*]$,
$$\dot{\mathbf{p}}^*(t) = -\frac{\partial \mathbf{v}_w}{\partial \mathbf{x}}(t, \mathbf{x}^*(t))^\intercal \, \mathbf{p}^*(t)$$

iii) *Transversality (free final time)*
$$\min_{\mathbf{s}\in\mathbb{S}^1} H(T^*, \mathbf{x}^*(T^*), \mathbf{p}^*(T^*), \lambda^*, \mathbf{s}) = 0$$

In our case, the Hamiltonian is smooth over the control variable so the point-wise minimization leads to

$$\text{For a.e. } t \in [0, T^*], \mathbf{s}^*(t) = -\frac{\mathbf{p}^*(t)}{|\mathbf{p}^*(t)|} \tag{5}$$

We define the augmented state $\mathbf{z} := \begin{pmatrix}\mathbf{x}\\ \mathbf{p}\end{pmatrix} = (\mathbf{x}^\intercal \ \mathbf{p}^\intercal)^\intercal$ and write its derivative:

$$\dot{\mathbf{z}}(t) = \begin{pmatrix}\dot{\mathbf{x}}(t)\\ \dot{\mathbf{p}}(t)\end{pmatrix} = \begin{pmatrix}-\mathrm{v}_a \frac{\mathbf{p}(t)}{|\mathbf{p}(t)|} + \mathbf{v}_w(t, \mathbf{x}(t))\\ -\frac{\partial \mathbf{v}_w}{\partial \mathbf{x}}(t, \mathbf{x}(t))^\intercal \, \mathbf{p}(t)\end{pmatrix} \tag{6}$$

With $\mathbf{x}_0$ fixed and $\mathbf{p}_0 \in \mathbb{R}^2$ a parameter, we note $\mathcal{S}[\mathbf{p}_0]$ the following Cauchy problem:

$$\mathcal{S}[\mathbf{p}_0] : \begin{cases} \mathbf{z}(0) = (\mathbf{x}_0^\intercal \ \mathbf{p}_0^\intercal)^\intercal \\ \mathbf{z}(\cdot) \text{ satisfies (6)} \end{cases} \tag{7}$$

which admits a unique solution, noted $\mathbf{z}_{\mathbf{p}_0}(\cdot)$ and called *augmented extremal trajectory*. We define $\pi_{\mathbf{x}}$ the projection to the state space, *i.e.* $\pi_{\mathbf{x}}(\mathbf{z}) = \mathbf{x}$ with $\mathbf{z} = (\mathbf{x}^\intercal \ \mathbf{p}^\intercal)^\intercal$. Then, $\pi_{\mathbf{x}}(\mathbf{z}_{\mathbf{p}_0}(\cdot))$ is the *extremal trajectory* associated to parameter $\mathbf{p}_0$. We now show an important property saying that the norm of $\mathbf{p}_0$ does not matter.

Invariance to scaling: For any $\alpha > 0$, $\pi_{\mathbf{x}}(\mathbf{z}_{\alpha\mathbf{p}_0}(\cdot)) = \pi_{\mathbf{x}}(\mathbf{z}_{\mathbf{p}_0}(\cdot))$

Proof. Let $\mathbf{z}_{\mathbf{p}_0}(\cdot) = (\mathbf{x}(\cdot)^\intercal \ \mathbf{p}(\cdot)^\intercal)^\intercal$. Define $\mathbf{z}'(\cdot) = (\mathbf{x}(\cdot)^\intercal \ \alpha\mathbf{p}(\cdot)^\intercal)^\intercal$ and notice that $\mathbf{z}'$ still satisfies (6) and $\mathbf{z}'(0) = (\mathbf{x}_0^\intercal \ \alpha\mathbf{p}_0^\intercal)^\intercal$. So, $\mathbf{z}'$ is the solution to $\mathcal{S}[\alpha\mathbf{p}_0]$. This solution is unique, thus $\mathbf{z}' = \mathbf{z}_{\alpha\mathbf{p}_0}$ which proves $\pi_{\mathbf{x}}(\mathbf{z}_{\alpha\mathbf{p}_0}(\cdot)) = \mathbf{x}(\cdot) = \pi_{\mathbf{x}}(\mathbf{z}_{\mathbf{p}_0}(\cdot))$.

In the PMP, there exist two kinds of extremal trajectories: normal ones, for which $\lambda = 1$, and abnormal ones for which $\lambda = 0$. The former are rather the norm while the latter are the exception. Abnormal extremal trajectories are not fundamental to establish the resolution method presented in Section IV so they are not discussed in this article. For normal ones, the transversality condition writes:

$$|\mathbf{p}^*(T^*)| \underbrace{(\mathrm{v}_a + \mathbf{s}^*(T^*)^\intercal \, \mathbf{v}_w(T^*, \mathbf{x}^*(T^*)))}_{\mathbf{s}^*(T^*)^\intercal \, \dot{\mathbf{x}}^*(T^*)} = 1 \tag{8}$$

So if we shoot an augmented extremal $\mathbf{z}_{\mathbf{p}_0}(\cdot) = (\mathbf{x}(\cdot)^\intercal \ \mathbf{p}(\cdot)^\intercal)^\intercal$ over a time-window $[0, T]$ and if we have $\mathbf{s}(T)^\intercal \, \dot{\mathbf{x}}(T) > 0$, then we can multiply $\mathbf{p}(\cdot)$ by a positive factor to satisfy the transversality (8) without changing the resulting $\mathbf{x}(\cdot)$ according to the invariance in scaling. This trajectory then satisfies all conditions of the PMP and is thus a valid candidate to optimality. Let us now do the following remarks:

i) In the steady case (wind field invariant in time), it can be shown that the sign of $\mathbf{s}^\intercal \, \dot{\mathbf{x}}$ is invariant in time and thus is the same as $\mathbf{s}(0)^\intercal \, \dot{\mathbf{x}}(0)$. So it is known before shooting whether we shoot a valid candidate to optimality or not.

ii) In general, the sign of $\mathbf{s}(T)^\intercal \, \dot{\mathbf{x}}(T)$ is not known in advance. So we will shoot extremal trajectories and check afterwards for which time stamps t they are valid candidates to optimality to go from $\mathbf{x}_0$ to $\mathbf{x}(t)$ by checking $\mathbf{s}(t)^\intercal \, \dot{\mathbf{x}}(t) > 0$. In practice, the opposite case rarely happens (when $\mathbf{s}(t)^\intercal \, \dot{\mathbf{x}}(t) < 0$ the vehicle is heading backwards from its trajectory).

So in what follows, we will look for the solution of the TNP by shooting extremal trajectories sampled for $\mathbf{p}_0 \in \mathbb{S}^1$. We call *extremal field* $\Phi_{\mathbf{x}_0}(T)$ the collection of all extremal trajectories over a time window:

$$\Phi_{\mathbf{x}_0}(T) := \bigcup_{\mathbf{p}_0\in\mathbb{S}^1} \{\pi_{\mathbf{x}}(\mathbf{z}_{\mathbf{p}_0}(t))| \ t \in [0, T]\} \tag{9}$$

We are specifically interested in tracking the border of the extremal field $\partial\Phi_{\mathbf{x}_0}(T)$ which we call *extremal front*.

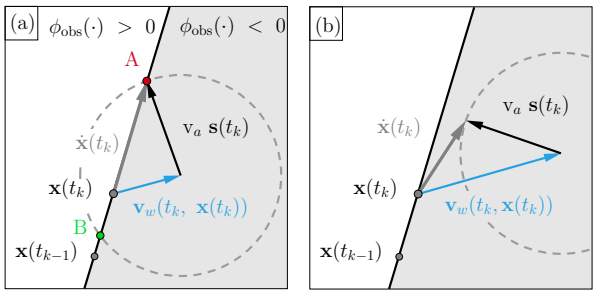

Fig. 2. State evolution in obstacle mode. Left, the determination of the appropriate heading vector to follow obstacle. Right, a situation where the wind is too strong and forces the integration to stop. The obstacle function being smooth and the analysis being conducted at infinitesimal scale, the obstacle boundary is drawn as a straight line.

C. Obstacles

Real problems always come with bounds or zones to avoid *e.g.* storms or restricted zones. State constraints are challenging for the application of the PMP. Still, some work has been successful for Zermelo's problem with bounds [3]. In a similar fashion, we derive optimality conditions in the presence of still obstacles by modifying the evolution of the augmented extremal. We characterize an obstacle by a differentiable obstacle function $\phi_{\text{obs}} : \mathbb{R}^2 \to \mathbb{R}$ for which $\{\mathbf{x} \in \mathbb{R}^2 | \ \phi_{\text{obs}}(x) < 0\}$ is the inside of the obstacle, $\{\mathbf{x} \in \mathbb{R}^2 | \ \phi_{\text{obs}}(x) > 0\}$ the outside and $\{\mathbf{x} \in \mathbb{R}^2 | \ \phi_{\text{obs}}(x) = 0\}$ its boundary.

The modified augmented extremal $\tilde{\mathbf{z}} = (\tilde{\mathbf{x}}^\intercal \ \tilde{\mathbf{p}}^\intercal)^\intercal$ in the presence of obstacles evolves likewise:

Rule for integration in the presence of obstacles:

i) When $\phi_{\text{obs}}(\tilde{\mathbf{x}}(t)) > 0$, using (6).

ii) When $\phi_{\text{obs}}(\tilde{\mathbf{x}}(t_{\text{obs}})) = 0$, using $\phi_{\text{obs}}^{-1}(\{0\})$ as ground route for $t \geq t_{\text{obs}}$. In general, two heading vectors are possible to follow the boundary (points A and B on Fig. 2). We choose the one which is consistent with the direction of the path and maximizes the absolute ground speed. If at a future time stamp the boundary becomes unflyable, *i.e.* the wind forces the trajectory either inside or outside the obstacle, then stop integration[1].

We now have all the theoretical ingredients to define a resolution method for the TNP.

IV. Method

A. Algorithm

We present a resolution method for the TNP in Algorithm 1. The general idea is similar to [9], [12]. We try here to exhibit a concise formulation of the procedure, and include the explicit handling of obstacles. Fig. 3 illustrates the following explanations.

We form an initial set of extremal trajectories using an uniform sampling of $\mathbb{S}^1$ for the value of $\mathbf{p}_0$, so they can cover uniformly the state space around $\mathbf{x}_0$, at least close to start. Then, the integration loop begins, applying any appropriate integration scheme (explicit Euler for instance) and using the rule defined in Section III-C to draw the trajectories. At some further time stamps, initial trajectories diverge from one another, so to keep precision under a given threshold, we need to shoot a new extremal trajectory between diverging ones. If $\mathbf{z}_1$ and $\mathbf{z}_2$ are current neighboring trajectories at t_0 which are diverging ($|\mathbf{x}_1(t_0) - \mathbf{x}_2(t_0)| > \varepsilon$) and their initial adjoint state is resp. $\mathbf{p}_0^{(1)}$ and $\mathbf{p}_0^{(2)}$, an intuitive idea would be to shoot the extremal trajectory $\tilde{\mathbf{z}}$ associated to $\frac{\mathbf{p}_0^{(1)}+\mathbf{p}_0^{(2)}}{2}$. But the latter would be very close to $\mathbf{z}_1$ and $\mathbf{z}_2$ over the whole time window $[0, t_0]$ and its integration up to t_0 would add no useful information to find the optimum. So the idea is instead to add directly a new extremal trajectory $\mathbf{z}'$ as "child" of $\mathbf{z}_1$ and $\mathbf{z}_2$ initialized with $\mathbf{z}'(t_0) = \left(\frac{\mathbf{x}_1+\mathbf{x}_2}{2} \;\; \frac{\mathbf{p}_1+\mathbf{p}_2}{2}\right)^\intercal$. Provided ε is sufficiently small, $\mathbf{z}'(t_0)$ is a good approximation of $\tilde{\mathbf{z}}(t_0)$ (we can show that when $|\mathbf{x}_1(t_0) - \mathbf{x}_2(t_0)| = O(\varepsilon)$, we have $|\mathbf{z}'(t_0) - \tilde{\mathbf{z}}(t_0)| = O(\varepsilon^2)$). This way, the collection of extremal trajectories shot by the procedure admits a *tree structure*.

The previous resampling scheme can be directly applied to trajectories evolving within obstacles and is even the key to track efficiently the extremal field around obstacles, when two neighboring particle are alternatively within and outside an obstacle.

We explain here two constructions appearing in the pseudo-code of Algorithm 1.

[1]When the wind forces the trajectory outside the obstacle, it is possible to resume shooting in mode i) by modifying the adjoint state with an additional Lagrange multiplier, as described in [3]. But for the present algorithm, the resampling scheme already approximates such trajectories with arbitrary precision (depending on the neighboring threshold ε and the time step dt), so we do not need to implement the additional Lagrange multiplier and we can simply stop trajectories unable to follow obstacle boundaries.

Data: $\mathbf{x}_0, \mathbf{x}_t, \mathbf{v}_w, \phi_{\text{obs}}$
Parameters: $N_{disc}, dt, \varepsilon$
Result: $T^*, \mathbf{z}^*$ ▷ Minimum time and optimal trajectory
$l \leftarrow [\,]$;
for $k \in \{0, 1, ..., N_{disc} - 1\}$ **do**
  $\mathbf{p}_0 \leftarrow \left(\cos\left(\frac{2\pi k}{N_{disc}}\right) \;\; \sin\left(\frac{2\pi k}{N_{disc}}\right)\right)^\intercal$;
  l.push(Traj$(\mathbf{x}_0, \mathbf{p}_0)$)
end
$\mathcal{Z} \leftarrow$ CyclicalGraph(l);
$t \leftarrow 0$;
while $True$ **do**
  $t \leftarrow t + dt$;
  for $\mathbf{z} \in \mathcal{Z}$ **do**
    $\mathbf{x}, \mathbf{p} \leftarrow \mathbf{z}$.tail();
    $\mathbf{x}_{\text{new}}, \mathbf{p}_{\text{new}}$, status $\leftarrow$ SingleStep$(\mathbf{x}, \mathbf{p})$;
    if status $= False$ **then**
      $\mathcal{Z}$.remove$(\mathbf{z})$; ▷ Cannot follow obstacle
    else
      $\mathbf{z}$.push$((\mathbf{x}_{\text{new}}^\intercal \;\; \mathbf{p}_{\text{new}}^\intercal)^\intercal)$;
      if $|\mathbf{x}_{new} - \mathbf{x}_t| < \varepsilon$ **then**
        return $t, \mathbf{z}$
      end
    end
  end
  for $\mathbf{z}_1, \mathbf{z}_2 \in \mathcal{Z}$.neighbors() **do**
    $\mathbf{x}_1, \mathbf{p}_1 \leftarrow \mathbf{z}_1$.tail(); $\mathbf{x}_2, \mathbf{p}_2 \leftarrow \mathbf{z}_2$.tail();
    if $|\mathbf{x}_1 - \mathbf{x}_2| > \varepsilon$ **then**
      $\mathcal{Z}$.addBetween$(\mathbf{z}_1, \mathbf{z}_2,$ Traj $\left(\frac{\mathbf{x}_1+\mathbf{x}_2}{2}, \frac{\mathbf{p}_1+\mathbf{p}_2}{2}\right))$
    end
  end
  [refinements]
end

Algorithm 1: Resolution of the TNP using extremal trajectories.

- "Traj" is an appropriate data structure to store a sequence of state and adjoint vectors representing an augmented extremal trajectory, with dedicated methods to get last element ("tail") and add an element ("push"). The structure is initialized with initial state and initial adjoint vectors.
- "SingleStep" applies the integration scheme on the kinematics of the vehicle, and complies with the rule defined in Section III-C. It has a flag "status" to signal that the integration has stopped: the only possible next step falls into an obstacle, whatever the control input.

Refinements: The algorithm involves creating new extremal trajectories along the way, which number may grow exponentially in some cases (highly unsteady winds for instance). But every trajectory is not needed to find the optimum. One may then add a *trimming* procedure at marker [refinements] in the algorithm to stop the integration of suboptimal trajectories based on a suboptimality detection criterion.

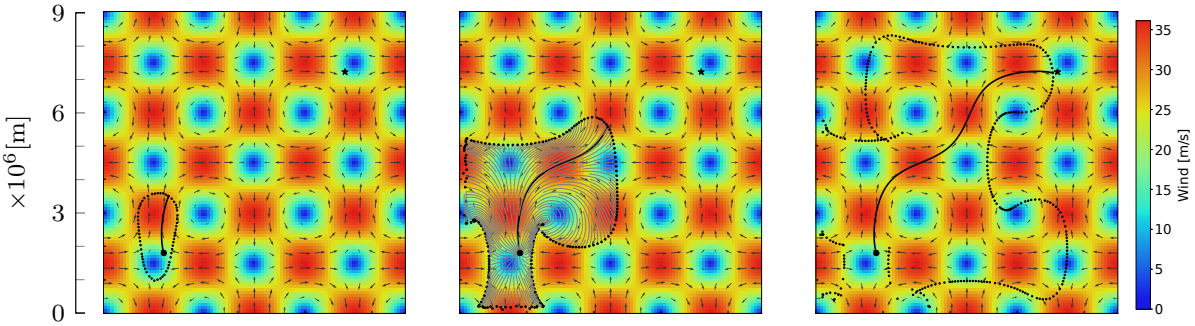

Fig. 3. The extremal front $\partial\Phi_{\mathbf{x}_0}$ is approximated by extremal trajectories (black points) for $t_1 = 10\,\mathrm{h}$, $t_2 = 30\,\mathrm{h}$ and $t_3 = 48\,\mathrm{h}\ 44\,\mathrm{min}$ (target reached). At t_2, the extremal field $\Phi_{\mathbf{x}_0}(t_2)$ is also represented, showing the resampling scheme and the collision filter at ridges. The fastest trajectory is drawn in black.

In [12], this is done by cutting off cycles occurring in the extremal pseudo-front (*i.e.* the closed curve $\bigcup_{\mathbf{p}_0\in\mathbb{S}^1}\pi_{\mathbf{x}}(\mathbf{z}_{\mathbf{p}_0}(t))$ at some time t). Still, this scheme is challenged by the presence of explicit obstacles which may cut a single extremal pseudo-front into independent sub-pieces. In this work, we implement a similar cycle-cutting scheme heuristic, adapted to the presence of obstacles[2].

In [9], a minimum-time mesh is built to keep track of the minimum time to reach every mesh point. This way, one may exclude extremal trajectories entering zones already visited by others. Still, this procedure is not valid for highly unsteady winds which may make it unavoidable to pass several times by the same region to reach destination (wind forces the path). We chose to implement a *collision filtering* scheme, which is similar to the previous idea, but only for steady problems. The idea is to prohibit the crossing of two extremal trajectories, which in the steady case implies that at least one of the two is suboptimal. The implementation is made efficient by updating a collision buffer dividing space in regular cells that keep track of possible conflicts in the corresponding area.

B. Implementation

The previous algorithm idea is implemented in a Python module called DABRY which source is made open[3]. The module aims at providing a complete pipeline to solve trajectory planning problems in flow fields. It handles analytically defined flow fields (hard-coded in Python using classes) as well as database wind (`grib` weather files). It features the previous extremal field algorithm (`solver_ef.py`). An interface to the Matlab® solver ToolboxLS[4] is provided for comparison to level-set methods. DABRY deals with planar as well as spherical problems, steady flow fields as well as dynamic ones.

V. EXPERIMENTS

A. Planar, steady but strong flow

We address the well known case of the analytical double gyre flow as benchmark for a strong, spatially non-uniform flow. We adapt the magnitude of the problem to the relevant scale for the Mermoz drone, *i.e.* thousands of kilometers as space scale and around $23\,\mathrm{m/s}$ for the flow. The problem data is

- $\mathbf{x}_0 = \begin{pmatrix}1.8\times10^6\mathrm{m} & 1.8\times10^6\,\mathrm{m}\end{pmatrix}^\intercal$
- $\mathbf{x}_t = \begin{pmatrix}7.2\times10^6\mathrm{m} & 7.2\times10^6\,\mathrm{m}\end{pmatrix}^\intercal$
- $\mathrm{v}_a = 23\,\mathrm{m/s}$
- $\mathbf{v}_w : (x_1, x_2) \mapsto \mathrm{v}_w \begin{pmatrix} -\sin(k_1\,(x_1 - x_1^{(c)}))\cos(k_2\,(x_2 - x_2^{(c)})) \\ \cos(k_1\,(x_1 - x_1^{(c)}))\sin(k_2\,(x_2 - x_2^{(c)})) \end{pmatrix}$

  $\mathrm{v}_w = 36.11\mathrm{m/s}$, $k_1 = k_2 = 1.04\times10^{-6}\,\mathrm{m}^{-1}$

  $x_1^{(c)} = x_2^{(c)} = 1.5\times10^6\,\mathrm{m}$

The solver finds a minimum travel time $T^* = 48\,\mathrm{h}\ 44\,\mathrm{min}$ in about $10\,\mathrm{s}$ of computation[5]. Fig. 3 shows the step-by-step computation where we can see the extremal front evolve. The optimal trajectory uses strong flow regions $|\mathbf{v}_w(\mathbf{x})| > \mathrm{v}_a$. The results, consistent with [7], were validated using ToolboxLS.

In reality, the output of the method is not just a single optimal trajectory, but a collection of extremal trajectories, as is depicted in Fig. 3. Many of these extremal trajectories provide the optimal travel time to other points in space: this happens for each one lying on the extremal front $\partial\Phi_{\mathbf{x}_0}(t)$. So the method really computes an optimal synthesis of the control problem. Level set methods also provide this complete resolution of the control problem, but one advantage of the extremal field method is the possibility to access optimal trajectories directly after computation, whereas for LSMs an additional shooting phase must be run, building the optimal control law from vectors that are normal to fronts.

This example thus illustrates how the algorithm builds optimal trajectories in a steady but strong flow field.

B. Spherical, unsteady and constrained

We then demonstrate a real life application of the method. We consider re-analyses of the Global Forecast Model (GFS) to work on real wind data. The wind is the surface wind (1000 hPa) extracted from Sep. 29th 2021 00:00Z to Oct. 1st 2021 00:00Z with a 6 hour time step[6]. This time window

[2]Implemented in the `trim` method of the `SolverEF` class in `solver_ef.py` from the DABRY module, see Section IV-B.

[3]`https://github.com/dabry-route/dabry`

[4]`https://www.cs.ubc.ca/~mitchell/ToolboxLS/`

[5]Intel®Core™i5-10210U CPU @ 1.60GHz × 8

[6]`https://www.ncei.noaa.gov/products/weather-climate-models/global-forecast`

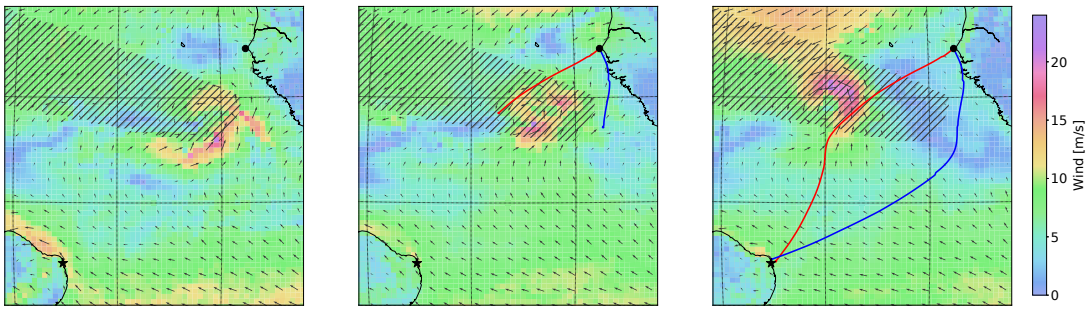

Fig. 4. Dakar to Natal crossing between Sep. 29th and Sep. 30th, 2021. The red trajectory is the optimal one in the absence of obstacles. The blue one takes a hashed no-go zone into account. The map is an orthographic projection of Earth centered at the middle point between Dakar and Natal (great circle). Tenth of degrees latitudes and longitudes are drawn in background. Color cells show wind field resolution of 0.5 degrees.

captures tropical storm "Victor" depicted in Fig. 4. The computation uses linear interpolation of the wind both in time and space and uses the spherical version of extremal evolution equations. Results are validated against ToolboxLS. The latter does not handle spherical geometry directly, so it is run on a projected wind field using an orthographic projection with the middle of the great circle between start and target as projection center. This projection is selected because it has few parameters (only its center) and it is a correct approximation for thousands of kilometers order of magnitude.

We run the computation unconstrained first, producing the red trajectory in Fig. 4, which arrives in $38\,\mathrm{h}$. This trajectory benefits from the advection of the high wind currents around the storm. By comparison, following the great circle ground route takes $36\,\mathrm{h}$ without wind and $51\,\mathrm{h}\ 49\,\mathrm{min}$ in this environment. This helps quantify the operational interest of optimal path planning in such a dynamic environment.

We can also imagine that it is not acceptable to get this close to storms. In Fig. 4, a no-go zone encapsulating the moving storm is displayed. We run again the computation taking this obstacle into account and get the blue trajectory reaching target in $43\,\mathrm{h}\ 30\,\mathrm{min}$.

Both computation were each run in less than $10\,\mathrm{s}$. This shows the efficiency of the method to deal with real-life examples featuring unsteady winds as well as the apparition of danger zones.

VI. Conclusion and Future Work

Time-optimal navigation in flow fields is of particular interest for slow airborne vehicles as well as underwater vehicles. In this paper, the relevance of extremal field methods has been demonstrated in an unsteady and constrained airborne time-optimal navigation problem.

To the best of the authors' knowledge, it is also the first demonstration of explicit obstacle handling in an operational example for EFM. More than 15 other examples are available in the DABRY module and displayed online[7].

The low computational cost of the method appeals generalization. In further work, we plan to study *energy-optimal* navigation in which the airspeed law is made variable. EFMs are indeed believed to keep the cost of computation low even for this larger problem, because the optimal airspeed law can also be found by shooting extremal trajectories.

Furthermore, it is planned to deal with uncertainty in the data to produce robust optimal paths. Weather ensemble forecast can be an input to a generalized EF method quantifying risk for trajectories.

[7] `https://dabry-route.github.io`

References

[1] S. J. Bijlsma. Optimal Aircraft Routing in General Wind Fields. *Journal of Guidance, Control, and Dynamics*, 32(3):1025–1029, May 2009.

[2] A. Chakrabarty and J. Langelaan. UAV flight path planning in time varying complex wind-fields. In *2013 American Control Conference*, pages 2568–2574, Washington, DC, June 2013. IEEE.

[3] R. Chertovskih, D. Karamzin, N. T. Khalil, and F. L. Pereira. An Indirect Method for Regular State-Constrained Optimal Control Problems in Flow Fields. *IEEE Transactions on Automatic Control*, 66(2):787–793, Feb. 2021.

[4] 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)*, pages 1403–1408. IEEE, 2014.

[5] D. González-Arribas, M. Soler, and M. Sanjurjo-Rivo. Robust Aircraft Trajectory Planning Under Wind Uncertainty Using Optimal Control. *Journal of Guidance, Control, and Dynamics*, 41(3):673–688, Mar. 2018.

[6] A. Guitart, D. Delahaye, and E. Feron. An Accelerated Dual Fast Marching Tree Applied to Emergency Geometric Trajectory Generation. *Aerospace*, 9(4):180, Mar. 2022.

[7] D. Kularatne, S. Bhattacharya, and M. A. Hsieh. Time and Energy Optimal Path Planning in General Flows, 2016.

[8] T. Lolla, P. F. J. Lermusiaux, M. P. Ueckermann, and P. J. Haley. Time-optimal path planning in dynamic flows using level set equations: theory and schemes. *Ocean Dynamics*, 64(10):1373–1397, Oct. 2014.

[9] A. Marchidan and E. Bakolas. Numerical Techniques for Minimum-Time Routing on Sphere with Realistic Winds. *Journal of Guidance, Control, and Dynamics*, 39(1):188–193, Jan. 2016.

[10] C. Petres, Y. Pailhas, P. Patron, Y. Petillot, J. Evans, and D. Lane. Path Planning for Autonomous Underwater Vehicles. *IEEE Transactions on Robotics*, 23(2):331–341, Apr. 2007.

[11] D. Rao and S. B. Williams. Large-scale path planning for Underwater Gliders in ocean currents. *Australasian Conference on Robotics and Automation*, 2009.

[12] B. Rhoads, I. Mezić, and A. C. Poje. Minimum time heading control of underpowered vehicles in time-varying ocean currents. *Ocean Engineering*, 66:12–31, July 2013.

[13] D. N. Subramani, Q. J. Wei, and P. F. Lermusiaux. Stochastic time-optimal path-planning in uncertain, strong, and dynamic flows. *Computer Methods in Applied Mechanics and Engineering*, 333:218–237, May 2018.