Generating Optimal Aircraft Trajectories with respect to Weather Conditions

Brunilde GIRARDET [a,1], Laurent LAPASSET [b], Daniel DELAHAYE [a], Christophe RABUT [c], and Yohann BRENIER [a]

[a] *ENAC-MAIAA, F-31055 Toulouse, FRANCE*

[b] *Capgemini, 15 avenue du Docteur Grynfogel, 31036 Toulouse Cedex1, FRANCE*

[c] *Université de Toulouse (INSA, IMT, MAIAA); 135, Av. de Rangueil, F-31077 Toulouse*

Abstract. Two major projects have been initiated to improve air traffic management by enabling 4D trajectory planning, whereby the aircraft plan their trajectory both in position and in time. In this paper, we are interested in a Free Flight variant of the concept, whereby airspace users are allowed maximum freedom when selecting routes: aircraft are no longer restricted to fly along airways; rather, they are allowed to fly along optimal routes, from origin to destination, following optimal altitudes, using favourable winds and avoiding hazards. Such optimal routes are good for the environment, for the airlines, and for passengers.

The goal of our research is to generate trajectories which minimize congestion and travel time of each aircraft in a way that is fair and efficient. We first optimize the route of a single aircraft relying on an algorithm called Ordered Upwind. And then, with a multi-agent system, we modify trajectories in order to minimize the congestion and to stay as close as possible to the optimal trajectories.

Keywords. Weather, Trajectory planning, Ordered Upwind Method, Multi-agent System

1. Introduction

1.1. Free-Flight Context

The *Free-Flight* concept [1] has been introduced in two major projects for air-traffic control: SESAR[2] in Europe and NextGen[3] in the United States. These projects have been initiated to improve Air Traffic Management (ATM) so as to be able to cope with the expected growth of air traffic for the next 20 years. The *Free-Flight* concept is based on the notion of 4D trajectory planning, whereby aircraft will plan their trajectories both in position and in time. To increase air traffic, airspace users will be allowed maximum freedom when selecting routes, then aircraft will be no longer restricted to fly along airways; rather, they will be allowed to fly along optimal routes, from origin to destination, following optimal altitudes, using favourable winds and avoiding hazards. In this context,

[1] Corresponding Author: Brunilde Girardet, ENAC-MAIAA, F-31055 Toulouse, FRANCE; E-mail: girardet@recherche.enac.fr

[2] Single European Sky Air traffic management Research

[3] Next Generation Air Transportation System

taking into account weather conditions, and especially wind, presents many advantages in terms of fuel consumption and travel time. The benefits to fly along wind-optimal routes have been studied in [2].

Removing fixed routes makes the traffic less predictable and increases the complexity of conflict solving, so special attention must be paid during trajectory planning to keep congestion below a safe level.

1.2. Related Work

Planning optimal trajectories is a rich and dynamic research domain with many application areas like robotics or space. Depending on the problem needs, the issues are different in nature and so are the techniques used to solve it. In ATM, optimality is defined using a composite criterion with a term directly related to fuel consumption and a second one linked with the congestion encountered along the flight path. A specificity of the problem is the fact that optimality with respect to fuel consumption has to deal with the wind field in the airspace, thus introducing an anisotropic criterion.

In [3], Jardin began to work on optimal aircraft trajectories in wind for some specific wind fields. He modelized his problem as an optimal control problem and solved it using an indirect method. In [4], Ng extended Jardin's work and developed a trajectory optimization algorithm for minimizing aircraft travel time and fuel burn by combining a method for computing minimum-time routes in winds on multiple horizontal planes, and an aircraft fuel burn model for generating fuel-optimal vertical profiles. The main drawback in using indirect methods is the need to find a good initial condition to start the algorithm.

Another method to address optimal control problems is to solve the Hamilton-Jacobi equation. In [5], Sethian and Valdirminsky introduced a new fast method, called Ordered Upwind Method, for computing approximate solutions to a wide class of static Hamilton-Jacobi equations with Dirichlet boundary conditions. This method is based on the previous work of Sethian in [6] on the Fast Marching Method, that has been developed to solve the Eikonal equation (the special case of Hamilton-Jacobi equations in which the problem is isotropic).

In [7], Alton used the Ordered Upwind method with the Semi-Lagrangian method to generate optimal trajectories. The drawback of this method is the need for a local minimization at each mesh point, which increases the computational time.

All of these methods are developed to optimize a single trajectory. In [7], Alton used his method to compute optimal trajectories for two robots without collision. The computational cost scales as a power of the number of conflicting robots and becomes prohibitive even for small instances of the problem: taking into account the performances of today computers, it is very difficult to plan more than two coordinated robots. With this constraint, we look for another approach to deal with several trajectories while taking advantage of the knowledge of the optimal trajectory for one aircraft. Multi-Agent Systems seem adapted to work with large and complex systems.

To be able to cope with the expected growth of air traffic, decentralized approaches are studied to handle air traffic; there would then no longer be a centralized control at ground to handle the air traffic, and each aircraft would be responsible for its own trajectory.

In [8], Sislak studied such a decentralized air traffic planning and control to provide a more efficient use of available airspace and to improve support for replanning and

collision avoidance. Each aircraft is able to plan its own trajectory, but also to detect and solve conflicts with other aircraft. Each aircraft is controlled by an agent and the agents are able to communicate with each other. Resolutions are performed in real-time; the aircraft are flying when the agents solve a conflict. The agents can make use of all degrees of freedom provided by flight dynamics: heading, altitude and cruise speed of the aircraft can be modified.

In [9], Agogino also uses a multi-agent algorithm to reduce congestion. Unlike Sislak's work, one agent does not control one aircraft. An agent is associated to a fixed (specific) location in order to decrease the congestion around this location. Each agent controls aircraft around him with one of these three actions: setting separation between aircraft, ordering ground delays, or performing reroutes. To choose the best control, the agents use *reinforcement learning*.

In contrast to Sislak's and Agogino's work, we shall not use a multi-agent system to control aircraft in real-time. Our goal is to generate all the trajectories before take off in order to minimize time travel and congestion. We use the multi-agent system to modify the optimal trajectories of each aircraft in order to decrease congestion. Nevertheless, the multi-agent system principle involved is similar to Sislak's and Agogino's work.

1.3. Contributions

The main contribution in this paper is the introduction of a new model for the optimal path planning problem in order to use the Eulerian discretization in the Ordered Upwind Method. Usually, the drawback of this discretization is the computation of the roots of a non-linear equation. In our case, we shall show that the resolution is easy since the equation is quadratic.

We introduce also our future work on coordinated trajectories. From the knowledge of the optimal trajectory for one aircraft, we look for a model based on multi-agent systems to deal with several trajectories.

1.4. Overview of the Paper

This article is organized as follows. Section 2 describes how one optimal aircraft trajectory is generated with respect to weather conditions. We present how we write our problem as a Hamilton-Jacobi equation. Then, we review the Ordered Upwind method to solve the Hamilton-Jacobi equation and we present results using this algorithm. Section 3 presents our future works on how we deal with the optimal trajectories of each aircraft to minimize congested areas.

2. Generating One Optimal Trajectory

2.1. Model

2.1.1. Aircraft Dynamics

As explained in Section 1.1, only en-route trajectories are concerned. We assume that aircraft fly at a constant flight level and at a constant True Airspeed. The True Airspeed

is based on BADA [4] database. Under these assumptions, the aircraft equations of motion are :

$$\begin{cases} \dot{x}(t) = V_a \cos(\theta(t)) + W_x(x,y) \\ \dot{y}(t) = V_a \sin(\theta(t)) + W_y(x,y) \end{cases} \tag{1}$$

with (x,y) the aircraft position, θ the heading angle, V_a the True Airspeed, $W_x(x,y)$ the east component of the wind and, $W_y(x,y)$ the north component of the wind.

2.1.2. Optimization Problem

Our goal is to compute heading along a trajectory yielding the minimal travel time between fixed origin and destination. Assuming constant True Airspeed and flight level, it turns out that the time optimal trajectory is also fuel optimal. Bolza formulation [10] of the problem is:

$$\begin{cases} u(x,y) = \min\limits_{\theta} & \int_{t_0}^{t_f} 1 dt \\ \text{s.t.} & \dot{x}(t) = V_a \cos(\theta(t)) + W_x \\ & \dot{y}(t) = V_a \sin(\theta(t)) + W_y \\ & (x(t_0), y(t_0)) = (x_0, y_0) \\ & (x(t_f), y(t_f)) = (x_f, y_f) \end{cases} \tag{2}$$

It is convenient to let the control variable be $\mathbf{a} = (\cos(\theta), \sin(\theta))$, a unit vector in the direction of motion of the aircraft. The optimization problem becomes:

$$\begin{cases} u(\mathbf{x}) = \min\limits_{\mathbf{a} \in \mathbb{A}} & \int_{t_0}^{t_f} 1 dt \\ \text{s.t.} & \dot{\mathbf{x}} = f(\mathbf{x}(t), \mathbf{a}(t)) \\ & \mathbf{x}(t_0) = \mathbf{x_0} \text{ and } \mathbf{x}(t_f) = \mathbf{x_f}, \end{cases} \tag{3}$$

with $\mathbf{x}$ the aircraft position and $f(\mathbf{x}(t), \mathbf{a}(t))$ the ground speed of the aircraft at time t. In Eq. (3), $u(\mathbf{x})$ represents the minimum time required to reach the point $\mathbf{x}$ starting from the initial point x_0.

2.2. The Hamilton-Jacobi Equation

Our optimal control problem (3) can be written as a Hamilton-Jacobi equation. The steps of the calculations can be found in [10]. The minimal-time optimal trajectory problem for continuous problems is the solution of:

$$\max_{\mathbf{a} \in \mathbb{A}} \{\nabla u(\mathbf{x}).f(\mathbf{x}, \mathbf{a})\} = 1 \tag{4}$$

[4] Base of Aircraft DAta, http://www.eurocontrol.int/services/bada

Eq. (4) can be rewritten as :

$$\|\nabla u(\mathbf{x})\| \max_{\mathbf{a}\in\mathbb{A}} \left\{ \frac{\nabla u(\mathbf{x})}{\|\nabla u(\mathbf{x})\|} . f(\mathbf{x},\mathbf{a}) \right\} = 1 \tag{5}$$

Eq. (5) shows a correspondence between the optimal trajectory problem and a wavefront propagation. The evolution of the front is described by Eq. (6) :

$$\|\nabla u(\mathbf{x})\| F(\mathbf{x}, \frac{\nabla u(\mathbf{x})}{\|\nabla u(\mathbf{x})\|}) = 1 \qquad \text{with} \qquad F(\mathbf{x},\mathbf{n}) = \max_{\mathbf{a}\in\mathbb{A}} \{\mathbf{n}.f(\mathbf{x},\mathbf{a})\} \tag{6}$$

where $\mathbf{n}$ is the outward unit vector normal to the front at the point $\mathbf{x}$ and $F(\mathbf{x},\mathbf{n})$ is the front speed in the direction $\mathbf{n}$, $u(\mathbf{x})$ represents the time at which the front passes through the point $\mathbf{x}$. Eq. (6) gives a correspondence between the speed of the wavefront, $F(\mathbf{x},\mathbf{n})$, and the speed of the aircraft, $f(\mathbf{x},\mathbf{a})$.

Eq. (6) describes the evolution of the front only if the speed function, F, never changes sign, in which case the front crosses a given point only once. If F is strictly positive (resp. negative), then the front is expanding (resp. contacting).

In the isotropic case, F does not depend on the direction. Eq. (6) simplifies to:

$$\|\nabla u(\mathbf{x})\| F(\mathbf{x}) = 1 \tag{7}$$

Eq. 7 is known as the Eikonal equation. The method, called *Fast Marching* method, developed by Sethian in [6] is a very efficient way to solve the Eikonal equation.

2.3. Ordered Upwind Algorithm

The Ordered Upwind algorithm was developed by Sethian and Valdirminsky for approximating the solution of the Hamilton-Jacobi equations. It was first introduced in [5]. In [11], Sethian proved that the algorithm converges to the viscosity solution of the Hamilton-Jacobi equations, a weak solution of a partial differential equation (PDE).

The principle of the Ordered Upwind algorithm is to avoid iterations through a careful use of the information about the characteristic directions of the PDE. This principle makes the algorithm highly efficient. It exploits the fact that the value function, u, is strictly increasing along the characteristics. $u(\mathbf{x})$ is then constructed gradually using the previous $u(\mathbf{x})$ value along the characteristic. In general, characteristic directions of the PDE are not known in advance. The strength of the Ordered Upwind method is the ability to compute information about the characteristic as the solution is constructed. In our problem, the characteristics of Eq. (4) represent the optimal trajectories.

Contrary to the Fast Marching Method for isotropic problem, the characteristic and the gradient of u are not in the same direction. However, in [5], Sethian defined the maximum angle between the characteristic and the gradient of u. In the algorithm, this bound allows to select the mesh points used to compute the value $u(\mathbf{x})$ at a given point $\mathbf{x}$. This bound can be computed from the bounds of the wavefront speed. We note F_1 the lower bound and, F_2 the upper bound.

The numerical resolution of the Hamilton-Jacobi equation is similar to graph-search algorithms such as Dijkstra's algorithm. However, in opposition to graph-search algorithms, the Ordered Upwind method is consistent, since when the grid is refined, the

obtained solution converges towards the exact solution of the Hamilton-Jacobi equation. That the complexity of the method is $O(N \log N)$, with N the number of mesh points, the same as for graph algorithms.

To compute the value function, u, we consider an unstructured triangulated mesh. Let $(\mathbf{x}, \mathbf{x_j}, \mathbf{x_k})$ be a simplex, the value of $u(\mathbf{x})$ is computed from $u(\mathbf{x_j})$ and $u(\mathbf{x_k})$ if the characteristic for the mesh point $\mathbf{x}$ lies inside the simplex $(\mathbf{x}, \mathbf{x_j}, \mathbf{x_k})$. Let h be the maximum distance between two adjacent mesh points (i.e. if the mesh points $\mathbf{x_j}$ and $\mathbf{x_k}$ are adjacent, then $\left\|\mathbf{x_j} - \mathbf{x_k}\right\| \leqslant h$). All mesh points belong to one of these classes (Figure 1):

- *Accepted* is the set of mesh points where the function u has been computed and frozen.
- *Considered* is the set of mesh points where an estimate, v, of u has been computed (but not frozen).
- *Far* is the set of all other mesh points where an estimate, v, of u has not been computed yet.

Two other sets are also created :

- *AcceptedFront* is defined as a subset of *Accepted* mesh points, which are adjacent to some not-yet-accepted (i.e. *Considered*) mesh points.
- *AF* is defined as a set of line segments $[\mathbf{x_j}, \mathbf{x_k}]$, where $\mathbf{x_j}$ and $\mathbf{x_k}$ are adjacent mesh points on the *AcceptedFront* and $\mathbf{x_j}$ and $\mathbf{x_k}$ are adjacent to a *Considered* mesh point $\mathbf{x}$.

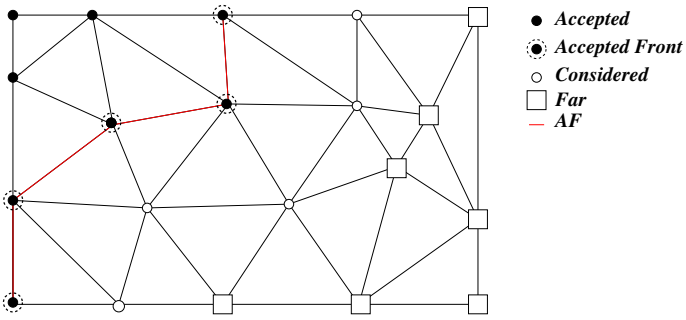

Figure 1. All the mesh points are assigned to three different sets: Accepted, Considered and Far. Accepted Front is a subset of the Accepted set. The AF set describes the front.

For each *Considered* mesh point $\mathbf{x}$, we define a new set called *NearFront*. It is a subset of *AF* segments, which are close to the Considered mesh point $\mathbf{x}$.

$$NF(\mathbf{x}) = \left\{ (\mathbf{x_j}, \mathbf{x_k}) \in AF | \exists \tilde{\mathbf{x}} \text{ on } (\mathbf{x_j}, \mathbf{x_k}) s.t. \|\tilde{\mathbf{x}} - \mathbf{x}\| \leqslant h \frac{F_2}{F_1} \right\}$$

From this discretization of the work space, we present the algorithm introduced in [11] to compute the propagation of a wavefront.

Ordered Upwind Algorithm

1. Start with all the mesh points in Far ($u = +\infty$);
2. Move the initial point $\mathbf{x_0}$ to Accepted ($u(\mathbf{x_0}) = 0$);

3. Move all the mesh points $\mathbf{x}$ adjacent to the initial point into Considered and evaluate the trial value $v(\mathbf{x})$ as:

$$v(\mathbf{x}) := \min_{\mathbf{x_i} \in NF(\mathbf{x})} v_{\mathbf{x_i}}(\mathbf{x}) \tag{8}$$

4. Find the mesh point $\bar{\mathbf{x}}$ with the smallest value of v among all Considered;
5. Move $\bar{\mathbf{x}}$ to Accepted ($u(\bar{\mathbf{x}}) = v(\bar{\mathbf{x}})$) and update the AcceptedFront;
6. Move the Far mesh points $\mathbf{x}$ adjacent to $\bar{\mathbf{x}}$ in Considered and compute their trial values by:

$$v(\mathbf{x}) := \min_{\mathbf{x_j x_k} \in NF(\mathbf{x})} v_{\mathbf{x_j x_k}}(\mathbf{x}) \tag{9}$$

7. Recompute the values for all the other Considered $\mathbf{x}$ such that $\bar{\mathbf{x}}\mathbf{x_i} \in NF(\mathbf{x})$ by:

$$v(\mathbf{x}) = \min \left\{ v(\mathbf{x}), \min_{\bar{\mathbf{x}}\mathbf{x_i} \in NF(\mathbf{x})} v_{\bar{\mathbf{x}},\mathbf{x_i}}(\mathbf{x}) \right\} \tag{10}$$

8. If Considered is not empty, then go to step 4.

There are two equivalent methods to compute the trial value $v(\mathbf{x})$ from a simplex $(\mathbf{x}, \mathbf{x_j}, \mathbf{x_k})$. The Semi-Lagrangian method requires performing a local minimization at each mesh point, whereas the finite-differences upwind update formula requires finding the roots of a non-linear equation. The semi-Lagrangian method is the most common method [7], [12]. But, the Eulerian discretization is used here since, with our model, an analytic solution of the discretized equation is found.

To compute the value $v_{\mathbf{x_j x_k}}(\mathbf{x})$, Eq. (6) is discretized using an upwind finite-difference discretization on a simplex $(\mathbf{x}, \mathbf{x_j}, \mathbf{x_k})$ such that the mesh points $\mathbf{x_j}$ and $\mathbf{x_k}$ are adjacent. The gradient, $\nabla u(\mathbf{x})$, is approached for the simplex $(\mathbf{x}, \mathbf{x_j}, \mathbf{x_k})$ with the approximation of the directional derivative of u in the directions defined by vectors P_j and P_k. The vectors P_j and P_k have to be linearly independent and are defined as :

$$P_j = \frac{\mathbf{x} - \mathbf{x_j}}{\|\mathbf{x} - \mathbf{x_j}\|} \qquad \text{and} \qquad P_k = \frac{\mathbf{x} - \mathbf{x_k}}{\|\mathbf{x} - \mathbf{x_k}\|} \tag{11}$$

We define the 2 by 2 non-singular matrix P having P_j and P_k as its rows.

Let $w(\mathbf{x})$ be the column vector of the directional derivatives of u in the directions P_j and P_k at the point $\mathbf{x}$. Then, we can compute the gradient $\nabla u(\mathbf{x})$ from the directional derivatives:

$$\nabla u(\mathbf{x}) = P^{-1} w(\mathbf{x}) \qquad \text{with} \qquad w(\mathbf{x}) = \begin{bmatrix} \frac{u - u_j}{\|\mathbf{x} - \mathbf{x_j}\|} \\ \frac{u - u_k}{\|\mathbf{x} - \mathbf{x_k}\|} \end{bmatrix} \tag{12}$$

We use a first-order discretization to approach the directional derivatives, $w(\mathbf{x})$, such that:

$$w(\mathbf{x}) \approx v_{\mathbf{x_j x_k}}(\mathbf{x})\alpha + \beta \qquad \text{with} \qquad \alpha = \begin{bmatrix} \frac{1}{\|\mathbf{x} - \mathbf{x_j}\|} \\ \frac{1}{\|\mathbf{x} - \mathbf{x_k}\|} \end{bmatrix} \text{ and } \beta = \begin{bmatrix} \frac{-u(\mathbf{x_j})}{\|\mathbf{x} - \mathbf{x_j}\|} \\ \frac{-u(\mathbf{x_k})}{\|\mathbf{x} - \mathbf{x_k}\|} \end{bmatrix} \tag{13}$$

The discretized equation is then:

$$\left\| P^{-1} w(\mathbf{x}) \right\| F(\mathbf{x}, \frac{P^{-1} w(\mathbf{x})}{\left\| P^{-1} w(\mathbf{x}) \right\|}) = 1 \tag{14}$$

Recall that $F(\mathbf{x},\mathbf{n}) = \max_{\mathbf{a}\in\mathbb{A}}\{\mathbf{n}.f(\mathbf{x},\mathbf{a})\}$ (see Section 2.2). To be able to solve analytically Eq. (14), we need to simplify the expression for the speed of the wavefront F. As the aircraft speed is $f(\mathbf{x},\mathbf{a}) = V_a\mathbf{a}+\mathbf{W}$, we can prove that in our case the speed of the wavefront is:

$$F(\mathbf{x},\mathbf{n}) = V_a + \langle\, \mathbf{n}, W \,\rangle \tag{15}$$

Figure 2 shows the geometric representation of the speed of the wavefront in function of the aircraft speed.

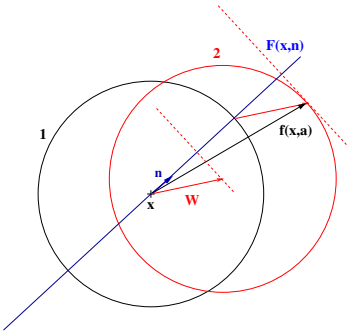

Figure 2. The figure shows how to compute the speed of the wavefront $F(\mathbf{x},\mathbf{n})$ in the normal direction $\mathbf{n}$ at the point $\mathbf{x}$. Circle 1 represents the true airspeed profile for the aircraft at the point $\mathbf{x}$ and circle 2 represents the speed of the aircraft profile $f(\mathbf{x},\mathbf{a})$ for all $\mathbf{a}$ at the point $\mathbf{x}$. The speed of the wavefront $F(\mathbf{x},\mathbf{n})$ is equal to the maximum of the projection of the aircraft's speed profile $f(\mathbf{x},\mathbf{a})$ on the direction normal $\mathbf{n}$.

From Eq.(15) and Eq.(14), we obtain:

$$\left\|P^{-1}w(\mathbf{x})\right\|^2 V_a^2 = \left(1-\langle P^{-1}w(\mathbf{x}),W\rangle\right)^2 \tag{16}$$

Eq. (16) is a quadratic equation of the form :

$$\begin{aligned} &Av^2_{\mathbf{x_j}\mathbf{x_k}}(\mathbf{x}) + Bv_{\mathbf{x_j}\mathbf{x_k}}(\mathbf{x}) + C = 0 \qquad \text{with} \qquad A = V_a^2\langle P^{-1}\alpha, P^{-1}\alpha\rangle - \langle P^{-1}\alpha, W\rangle^2 \\ &\qquad\qquad B = 2V_a^2\langle P^{-1}\alpha, P^{-1}\beta\rangle - 2\langle P^{-1}\alpha, W\rangle\left(\langle P^{-1}\beta, W\rangle - 1\right) \\ &\qquad\qquad C = V_a^2\langle P^{-1}\beta, P^{-1}\beta\rangle - \left[\langle P^{-1}\beta, W\rangle - 1\right]^2 \end{aligned} \tag{17}$$

We remind that the algorithm exploits the fact that the value function, u, is strictly increasing along the characteristics, in order to construct gradually the value function, $u(\mathbf{x})$, in function of the previous value function along the characteristic. To ensure that the value of $v_{\mathbf{x_j}\mathbf{x_k}}$ computed from (17) is a good approximation of the value function, u, at the point $\mathbf{x}$; the characteristic for the mesh point $\mathbf{x}$ needs to lie inside the simplex $(\mathbf{x},\mathbf{x_j},\mathbf{x_k})$ (cf Figure 3). Otherwise, the value $v_{\mathbf{x_j}\mathbf{x_k}}$ for the point $\mathbf{x}$ is unacceptable, it should be computed using another simplex.

To check that the characteristic for the mesh point $\mathbf{x}$ lies inside the simplex $(\mathbf{x},\mathbf{x_j},\mathbf{x_k})$, we need to compute an approximation of the characteristic direction. The characteristic direction is $V_a\frac{\nabla u}{\|\nabla u\|}+W$. Thus, we need to calculate the approximation of the gradient of u. It can be approached with : $\nabla u(\mathbf{x}) \approx P^{-1}\left(\alpha v_{\mathbf{x_j}\mathbf{x_k}}(\mathbf{x})+\beta\right)$. From the approximation of the characteristic at the point $\mathbf{x}$, we are able to check that it lies inside the simplex $(\mathbf{x},\mathbf{x_j},\mathbf{x_k})$. For that purpose, we work in the new basis generated by the vectors P_j,

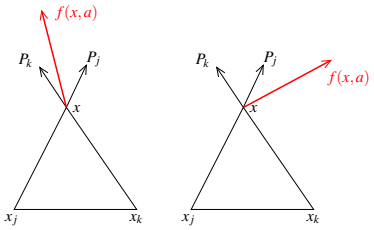

Figure 3. Example of acceptable (left) and unacceptable (right) approximations for $f(\mathbf{x},\mathbf{a})$, where the upwinding requirement is not satisfied and the update for the point $\mathbf{x}$ should be computed using others simplexes.

P_k. In this basis, the characteristic direction is $(P^T)^{-1}f(\mathbf{x},\mathbf{a})$. The upwind criterion is equivalent to the condition that all the elements of the characteristic direction in the basis (P_j,P_k) should be positive.

The drawback of this method is that it is based on the approximation and not the exact characteristic direction. One element of the characteristic direction can be close to zero. Due to the approximation, this element can be negative and the upwind criterion will not be satisfied, even if the true characteristic direction satisfies the criterion. This drawback can happen only when the characteristic direction is in the same direction as one of the vectors of the basis P_j or P_k, i.e. $v_{\mathbf{x_j x_k}}$ can be computed based on either $u(\mathbf{x_j})$ or $u(\mathbf{x_k})$. Finally, the value $v_{\mathbf{x_j x_k}}$ is computed as :

- if P_j and P_k are linearly independent and the upwind criteria is satisfied:
  $v_{\mathbf{x_j x_k}}$ = solution of the quadratic equation (17)
- otherwise: $v_{\mathbf{x_j x_k}} = \min(v_{\mathbf{x_j}}(\mathbf{x}), v_{\mathbf{x_k}}(\mathbf{x}))$ with $v_{x_i}(\mathbf{x}) = \frac{\|\mathbf{x}-\mathbf{x_i}\|}{f(\mathbf{x},\frac{\mathbf{x}-\mathbf{x_i}}{\|\mathbf{x}-\mathbf{x_i}\|})} + u(\mathbf{x_i})$

We have chosen to work with Eulerian discretization for our problem since the step to compute the trial value comes to solve a quadratic equation. Contrary to Semi-Lagrangian discretization, this step does not required iterative algorithms. However, for other problems, the equation will not necessarily be quadratic with Eulerian discretization, and it will require the computation of the roots of the non-linear equation with an iterative algorithm.

We can construct the optimal trajectory by tracing backward, from the arrival point to the initial point, and following the characteristic.

We propose to modify the speed of propagation. It is possible to slow down the wavefront in parts of the environment that have to be avoided. Thus, the value function, u, is increased, penalizing the passage through these areas.

2.4. Results

Our first test problem aims at emphasizing the impact of wind on trajectories. A model for obstacles has also been implemented. From the point of view of airlines' operations, these obstacles could model some adverse weather conditions such as storms or turbulences. In Figure 4, the optimal trajectories without wind (1.) and with wind (2.) have been computed using the Ordered Upwind algorithm with the Eulerian discretization. The obstacles are described by coloured iso-contours, red represents impassable obstacles and the colour goes toward the blue as the importance of the obstacles decreases. In

the north part, wind is in the east direction and increases northwards. In the south part, wind is in the west direction and increases southwards.

The trajectory taking into account the wind takes advantage of favourable winds. The two trajectories do not behave in the same way to get around the obstacles. This is attractive to get into account the wind to plan the aircraft trajectory. The wind-optimal trajectory can be far from the optimal trajectory computed without wind.

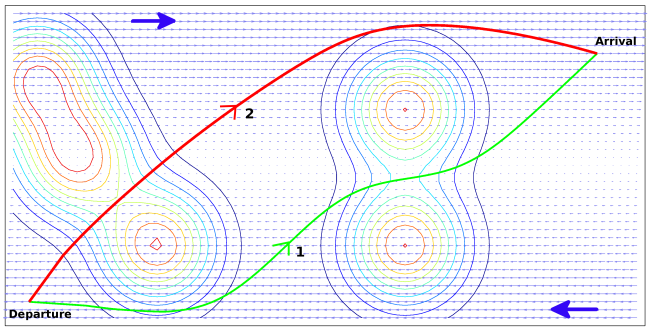

Figure 4. Optimal trajectory with obstacles: 1. Without wind (green), 2. With wind (red)

The second test problem aims at demonstrating the benefits of flying along optimal trajectories with a realistic wind model. We compare the travel time for the wind-optimal route with that of the direct route. As in the first test problem, the wind-optimal trajectory has been computed using the Ordered Upwind algorithm with the Eulerian discretization. In this example, the average wind is 56 kt. We chose 390 kt for the True Airspeed of the aircraft.

Figure 5 depicts both trajectories: the direct route (1.) and the wind-optimal trajectory (2.). Table 1 lists the travel time computed for both trajectories taking into account wind. For a flight time of about 25 minutes, the benefit of flying along the optimal route is 28 seconds. It represents 1.9% of time saved for the trajectory.

Table 1. Travel times for direct and optimal route.

	Travel time (seconds)
Direct route	1498
Optimal route	1470

2.5. Limits

In [7], Alton optimizes the trajectory of two robots without collision. He used the *Fast Marching* algorithm since he worked with an isotropic problems. Instead of solving the problem in $\mathbb{R}^2$, he worked in the configuration space of several robots. For two robots, the work space is $\mathbb{R}^4 - \Delta$, the (x,y) coordinates of both robots without the diagonal $\Delta = \{(x_1,y_1) = (x_2,y_2)\}$. By construction, the collision points between robots are excluded from the space. Computed trajectories are then ensured to be without collision. Space dimension is the main drawback of these algorithms since the state space increases exponentially with the number of mobiles involved in the problem. Resolution with more than two robots becomes computationally intractable.

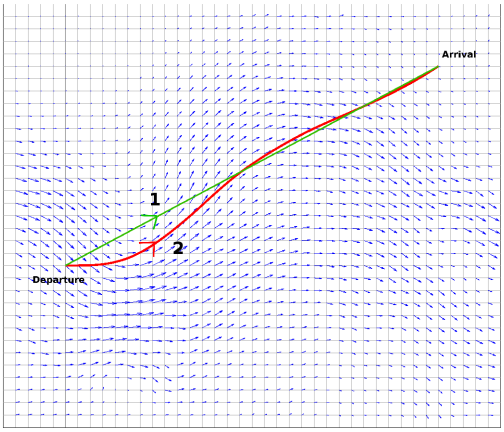

Figure 5. Trajectories: 1. Direct route (green), 2. Optimal route (red)

3. Future works : Multi-Agent Systems

Once the optimal trajectory of each aircraft is known (assuming the aircraft is the only one flying), our aim is to modify this trajectory in order to minimize congestion while remaining as close as possible to the optimal trajectories. Adaptive Multi-Agent Systems (AMAS) allow designers to focus more on local interactions rather than on the overall system [13]. From a set of local rules and cooperative behaviour of agents, Picard [14] emphasizes the emergence of global order from local considerations. In our multi-agent model, the agent-based trajectories are penalized or favoured close to the areas of congestion. The multi-agent system produces a map of penalized or favoured areas per flight taking into account wind and congestion. According to these maps, the wavefront is then slowed down or speeded up to make flight take advantage of wind by avoiding the congested areas. The main steps to compute the trajectories in the case of several aircraft are:

1. Compute optimal trajectory for each aircraft independently;
2. Detect congested areas;
3. Build the maps of penalization through the MAS in order to minimize induced congestion;
4. Compute again the optimal trajectories for each aircraft while penalizing or favouring areas.

The main issue of our approach is to create a map for each aircraft that indexes the congested areas. Each map is computed in order to further decrease the overall congestion and, the travel time for each flight. The use of the Ordered Upwind method described above allows us to generate a new optimal trajectory for each flight in relation to these maps which prevent the airspace from congestion creation.

To minimize congested areas and travel time, the penalized areas are computed in a coordinated manner between the flights. Here are the benefits of a multi-agent system. Introduced agents are: flights and waypoints. Flights are in charge of assessing the behaviour of their own waypoint mainly from a flight time perspective. One waypoint indexes one congested area for one flight. A waypoint agent is in charge of the filter aim-

ing at decreasing locally the congestion. Criticalities of the flights are the criteria which lead waypoints of a common congested area to take coordinated decisions. These waypoints decisions aim at leading related flight out of the congested area. The choices of a waypoint linked to a flight with high criticality will be more dominating in the MAS evolution.

4. Conclusion

We presented in this paper a new model for the optimal path planning problem of a single trajectory. We proposed an analytical solution of the discretized equation for the Ordered Upwind algorithm. Thus, this model has the advantage to avoid an iterative algorithm for the resolution of the equation. The results show the benefits of flying along optimal route with respect to weather conditions.

The next steps of this work will focus on developing the coordination between trajectories taking advantage of the knowledge of the optimal trajectories for each aircraft. The multi-agent model will be tested on large scale and realistic examples.

References

[1] R. L. Schultz, D. Daner, and Y. Zhao. Free-flight concept. In *Proceedings of the AIAA Conference on Guidance, Navigation, and Control*, pages 11–13, 2000.

[2] K. Palopo, R. D. Windhorst, S. Suharwardy, and H. Lee. Wind-optimal routing in the national airspace system. *Journal of Aircraft*, 47(5):1584–1592, 2010.

[3] M. Jardin and A. Bryson. Neighboring optimal aircraft guidance in winds. *Journal of Guidance, Control, and Dynamics*, 24(4):710–715, 2001.

[4] H.K. Ng, B. Sridhar, and S. Grabbe. A practical approach for optimizing aircraft trajectories in winds. In *Digital Avionics Systems Conference (DASC)*, volume 31, 2012.

[5] J.A. Sethian and A. Vladimirsky. Ordered upwind methods for static Hamilton-Jacobi equations. *Proceedings of the National Academy of Sciences*, 98(20):11069, 2001.

[6] J.A. Sethian. Fast Marching Methods. *SIAM review*, 41(2):199–235, 1999.

[7] K. Alton. *Dijkstra-like ordered upwind methods for solving static Hamilton-Jacobi equations*. PhD thesis, The University of British Columbia, 2010.

[8] David Sislak. *Agent-Based Approach to Air-Traffic Modeling, Simulation and Collision Avoidance*. PhD thesis, Czech Technical University, 2013.

[9] A. Agogino and K. Tumer. A multiagent approach to managing air traffic flow. *Autonomous Agents and Multi-Agent Systems*, 24(1):1–25, 2012.

[10] D. Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific Belmont, 1995.

[11] J.A. Sethian and A. Vladimirsky. Ordered upwind methods for static Hamilton-Jacobi equations: Theory and algorithms. *SIAM Journal on Numerical Analysis*, 41:325–363, 2003.

[12] J.S. Elston. *Semi-Autonomous Small Unmanned Aircraft Systems for Sampling Tornadic Supercell Thunderstorms*. PhD thesis, University of Colorado, 2012.

[13] J-P. Georgé, M-P. Gleizes, and V. Camps. Cooperation. In *Self-organising Software*, Natural Computing Series, pages 193–226. Springer, 2011.

[14] G. Picard. Agent Model Instantiation to Collective Robotics in ADELFE . In *Fifth Internatinal Workshop on Engineering Societies in the Agents World*, pages 209–221. Springer, octobre 2004.