Fourteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2021)

A Fast and Flexible Emergency Trajectory Generator

Enhancing Emergency Geometric Planning with Aircraft Dynamics

Raúl Sáez, Homeyra Khaledian, Xavier Prats
Department of Physics - Aerospace division
Technical University of Catalonia (UPC)
Castelldefels, Barcelona, Spain
{raul.saez.garcia, homeyra.khaledian,
xavier.prats}@upc.edu

Andréas Guitart, Daniel Delahaye
OPTIM Research Lab
Ecole Nationale de
l'Aviation Civile (ENAC)
Toulouse, France
andreas.guitart@enac.fr,
daniel.delahaye@recherche.enac.fr

Eric Feron
School of Aerospace Engineering
Georgia Institute of Technology
Atlanta, USA
eric.feron@aerospace.gatech.edu

***Abstract*—We present an automated emergency trajectory generator to compute the best emergency trajectory for a given landing site. A combination of the optimized version of the rapidly exploring random tree algorithm and Dubins paths is used to compute a path connecting the aircraft position with the landing site, which avoids the obstacles in the way. Then, this path is used as input for a trajectory prediction (TP) algorithm, which computes a four-dimensional trajectory by taking into account the current aircraft performance and weather. The set of vertical profiles considered in the TP has been designed in order to cover the widest possible range of emergency situations. Moreover, the aircraft intents considered in these profiles are chosen by taking into account the operational requirements of the air traffic operation system and the input of the flight crew. Among these profiles, two have been tested during the study, to verify the result of the proposed algorithm and its computing time, which is one of the main success criteria. This concept is expected to be part of an advanced flight management system on-board function to help the pilot take efficient and effective decisions in emergency situations and adverse conditions.** [1]

Keywords—Aircraft emergency planning; Safety; Trajectory prediction; Dubins paths; Path generation; Geometric route planning

I. Introduction

Nowadays, emergency trajectories for airliners in a degraded flyability mode do not exist except for engine loss situations in SIDs (standard instrumental departures), for which airliners have to specifically design the corresponding flight procedure. The current process of defining emergency trajectories and landing sites remains completely manual and fully relies on the capabilities of the pilot for situation analysis. In emergency situations, an automated support to the pilot could suppose a clear advantage by providing a trajectory to safely bring the aircraft from the location where the emergency takes place until a safe and appropriate landing site.

We can observe in history several cases of successful landings with very degraded aircraft capabilities—e.g., the U.S. flight 1549, ditching in the Hudson river after a double engine failure caused by bird strike—in which, thanks to a good situation analysis of the aircraft crew, the safety of the passengers and integrity of the aircraft structure were preserved. However, in other situations the consequences were fatal and, even if the outcome is positive in some cases, having an automated emergency trajectory generator could help to ensure avoiding compromising the safety of the operation in abnormal situations.

In this work, we present an emergency-trajectory-generation function for commercial aircraft allowing a safer return to ground when normal operation is interrupted, in the context of future automated operations with reduced crew. Emergency trajectories are generated by using a combination of the optimized version of the rapidly exploring random tree algorithm (RRT*) and Dubins paths, which are used to generate the route (i.e., lateral path), avoiding the obstacles present from the aircraft position to the landing site. Then, a four-dimensional (4D) trajectory is generated with a trajectory predictor (TP), where both the aircraft performance and the most updated weather forecast available are taken into account. A comprehensive set of vertical profiles—used when generating the 4D trajectory—have been identified in this work, designed to be as generic as possible in order to cover the widest possible range of emergency situations. Ultimately, the outcome of this work is to bring a support to the pilot—in both flight management and decision making—in emergency situations with degraded aircraft capabilities, where emergency trajectories would be injected within the flight management system (FMS). In this work, we focus only on the generation of the trajectory, assuming a proper landing site has been previously identified. In future work, an insight on the landing site selection process will be given, and a wider range of situations will be investigated in order to assess the feasibility

[1]The work presented in this paper has received funding from the Clean Sky 2 Joint Undertaking (JU) under grant agreement No 864771, corresponding to the SafeNcy project (https://cordis.europa.eu/project/id/864771). The JU receives support from the European Union's Horizon 2020 research and innovation programme and the Clean Sky 2 JU members other than the Union. The opinions expressed herein reflect the authors view only. Under no circumstances shall the Clean Sky 2 JU be responsible for any use that may be made of the information contained herein.

of our concept.

Similar works have dealt with the generation of emergency trajectories for civil aircraft. For instance, in [1], the authors presented an automatic post-failure flight planning to enable safe emergency landings. More recently, in [2], the authors also investigated the problem of the generation of a safe landing trajectory for an airplane with structural damage to its wing and flying very close to local terrain. In [3], an emergency planning technique focusing on small aircraft was presented. However, in all these works, only one type of emergency was considered (i.e., loss of thrust) and no obstacles were taken into account when generating the trajectory. Similarly, in [4], trajectories were generated for a loss of thrust emergency, but only considering purely geometric criteria for the generation of the trajectory, thus, neglecting aircraft dynamics. Finally, other remarkable works considering path planning under an emergency can be found in [5], where road-maps are used to generate the path, or in [6], where a real-time flight trajectory generator was presented. Furthermore, Garmin Ltd. developed the Autoland system [7], exclusively dedicated to general aviation. It consists in an autopilot-based functionality which, in case of an emergency where the pilot is unable to fly, determines the most suitable airport and runway and leads the aircraft to that runway, by taking into account weather, terrain, obstacles and aircraft performance capabilities.

To the best of the authors' knowledge, although there are many works dealing with the generation of emergency trajectories, they are mostly based on the geometric planning of these trajectories. Furthermore, most of the existing works do not take into account the operational requirements of the air traffic operations system (or the flight crew point of view), or they just compute trajectories for one specific emergency. In some cases, the existing solutions are only applied to general aviation. Finally, a lot of research is based on very simple scenarios, where no obstacles are found in the vicinity. The methodology proposed in this paper tries to bridge all these gaps, producing operationally sound trajectories tailored to specific aircraft (degraded) performance, considering weather and obstacle environments and covering a wide range of emergency or urgency situations.

II. Background

In this Section, we describe the different techniques we used in this work to generate the emergency trajectory. In Subsections II-A and II-B, we describe the methods used to generate the path: the RRT algorithm and Dubins path respectively. Then, in Subsection II-C, we focus on the trajectory prediction problem.

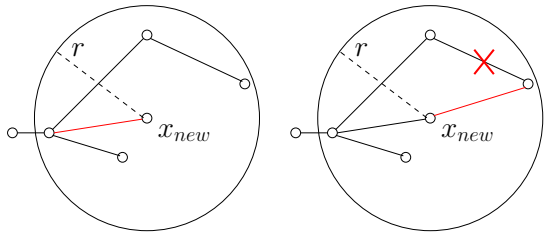
A. Path Planning Algorithm: Rapidly Exploring Random Tree Algorithm

Many methods propose to generate a graph in order to then find the optimal path between two points. The most used methods are sampling-based path planning algorithms, which consist in generating a graph from a sampling [8]. In this paper, we use one of these algorithms to generate the path, the RRT. Before discussing the algorithm, it is essential to define the problem and the main functions of this algorithm.

Let $\chi = (0,1)^d$ be the configuration space, where $d \in \mathbf{N}$ is the space dimension, ($d \geq 2$). Let χ_{obs} be an open set, which denotes the obstacle-free space as $\chi_{free} = cl(\chi \backslash \chi_{obs})$, where $cl(\chi)$ denotes the closure of a set χ. The initial condition is denoted by $x_{init} \in \chi_{free}$ and the goal region, χ_{goal}, is an open set of χ_{free}.

This algorithm is composed of four main functions. The first one is the sampling function, which generates a sequence of points in χ. The second one is the nearest neighbor function, which determines the vertex that is closest to a point $x \in \chi$. The third function is the near vertices function, which returns a set of vertices that are contained in a ball of radius r centered at a point $x \in \chi$. Finally, the last function is the collision test function, used to know whether the straight line between two points lies in χ_{free} or not.

At the beginning of this algorithm, the graph is composed of only the initial vertex and no edges. Then, at each iteration, a new point $x_{rand} \in \chi_{free}$ is generated. After that, this point is steered towards its nearest point. Given two points $x, y \in \chi$, the function $Steer : (x, y) \mapsto z$ returns a point $z \in \chi$ such that z is "closer" to y than x. The point returned by this function is connected to the graph if there is no obstacle between this point and the nearest point (Figure 1(a)). An improvement of such algorithm is the RRT* algorithm. It connects vertex in the same way as RRT. Moreover, at each step it tries to improve the graph around this new vertex. If the addition of the new vertex can reduce the cost of vertices that are within a distance r, a new connection replaces the previous connection (Figure 1(b)). The distance r is computed as a function of V, which is an open set composed of nodes, and a coefficient γ_{RRT*}, whose value depends on the search space.



(a) The new node x_{new} is connected to a neighboring node which minimizes its cost

(b) The cost of a neighbor is improved by the creation of x_{new}. The edge which connected this node to the graph is deleted and a new edge is created

Fig. 1. New connection building (a) and Connection replacement (b)

Figure 2 shows, for the same number of iterations, the graph generated by the RRT algorithm (left) and the RRT* algorithm (right). This figure shows that with the phase of tree improvement, the RRT* tree is more ordered than the RRT tree and, therefore, the path computed by the RRT* algorithm is closer to the shortest path.

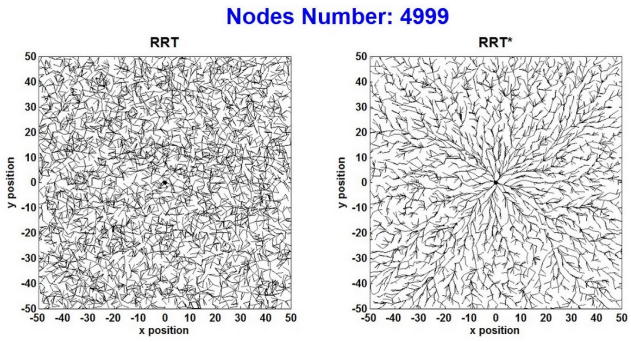


Fig. 2. Comparison between the graph generated by the RRT algorithm (left) and the one generated by the RRT* algorithm (right)

The RRT* is a very efficient algorithm because its time complexity is $O(n \log n)$. However, the generated paths are not flyable. Therefore, it is necessary to modify this algorithm in order to take into account the minimum turn radius and the flight path angles computed by the path bounds generator. The resulting trajectory will not be completely flyable, but it will ensure that all obstacles are avoided. Then, the motion planner (Section III-D) will take into account the aircraft performance to generate the flyable trajectory that the aircraft will follow.

B. Dubins Path

Dubins path is the shortest curve that connects two points with a constraint on the curvature of the path and with initial and final orientation angles. There are two different types of Dubins curve [9] [10]. The first one is the Circle-Segment-Circle (CSC), which is composed of a first turn, a segment, and a second turn. The second type of Dubins curve is Circle-Circle-Circle, which is composed of three different turns. In Sections II-B1 and II-B1, we show two examples of Dubins curves, a Left-Segment-Left Dubins curve and a Left-Right-Left Dubins curve respectively.

Let $P_1 = (x_1, x_2)$ and $P_2 = (x_2, y_2)$ be the initial position and the final position respectively. Let χ_1 and χ_2 be their track direction. An associated value of track is the orientation angle. Let α and β be the initial and final orientation angle defined as follows:

$$\begin{cases} \alpha = mod(\frac{\pi}{2} - \chi_1, 2\pi) \\ \beta = mod(\frac{\pi}{2} - \chi_2, 2\pi) \end{cases} \tag{1}$$

1) Left-Segment-Left: Figure 3 depicts a Left-Segment-Left Dubins curve. The curve connects a starting point with an orientation angle α and an end point with an orientation angle β. It is composed of a turn to the left around the first green circle, a segment of length L_s and a second turn to the left around the second green circle.

The length of the segment L_S and the two angles ϕ_1 and ϕ_2 are defined as follows :

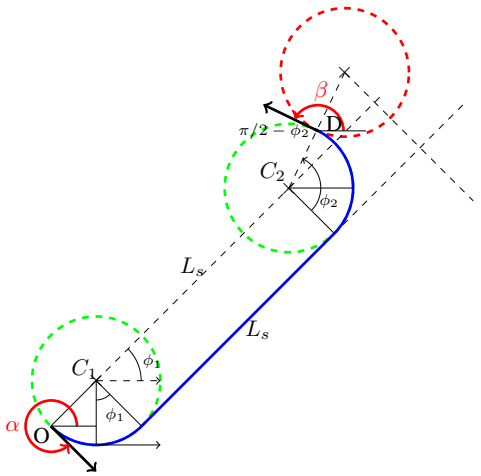


Fig. 3. Left-Segment-Left Dubins Curve

$$FirstTurnAngle = \begin{cases} \phi_1 - \alpha \ if \ \alpha \leq \phi_1 \\ \phi_1 + (2\pi - \alpha) \ otherwise \end{cases} \tag{2}$$

$$L_S = \sqrt{(x_2 - r\sin\beta - x_1 + r\sin\alpha)^2 + (y_2 + r\cos\beta - y_1 - r\cos\alpha)^2} \tag{3}$$

$$\phi_1 = mod(\arctan\left(\frac{y_2 + r\cos\beta - y_1 - r\cos\alpha}{x_2 - r\sin\beta - x_1 + r\sin\alpha}\right) \tag{4}$$

$$\phi_2 = mod(\beta - \phi_1, 2\pi) \tag{5}$$

2) Left-Right-Left: Figure 4 depicts a Left-Right-Left Dubins curve. The curve connects a start point with an orientation angle α and an end point with an orientation angle β. It is composed of 3 different turns.

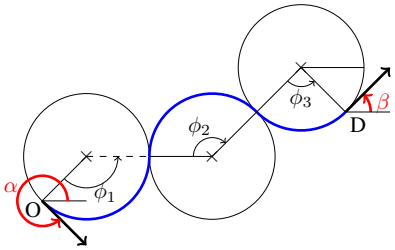


Fig. 4. Left-Right-Left Dubins curve

The three angles, ϕ_1, ϕ_2 and ϕ_3, are defined as follows:

$$\phi_1 = mod(-\alpha + \arctan\left(\frac{y_2 - y_1 - r\cos\alpha + r\cos\beta}{x_2 - x_1 + r\sin\alpha - r\sin\beta}\right) + \frac{\phi_2}{2}, 2\pi) \tag{6}$$

$$\phi_2 = \arccos\frac{1}{8r^2}(6r^2 - (y_2 - y_1)^2 - (x_2 - x_1)^2 + 2r^2\cos(\alpha - \beta) + 2(x_2 - x_1)r(\sin\alpha - \sin\beta) + 2(y_2 - y_1)r(\cos\beta - \cos\alpha)) \tag{7}$$

$$\phi_3 = mod(mod(\beta, 2\pi) - \alpha + 2\phi_2, 2\pi) \tag{8}$$

C. The Trajectory Prediction Problem

Let us divide the aircraft trajectory into N flight phases. For each phase i, defined over the time period $[t_0^{(i)}, t_f^{(i)}]$, a state vector $\boldsymbol{x}^{(i)}(t)$ and a control vector $\boldsymbol{u}^{(i)}(t)$ are defined. In this paper, the state vector $\boldsymbol{x} = [v, h, s, m]$ is composed, respectively, by the True Airspeed (TAS), the geometric altitude, the distance to go and the mass of the aircraft; while the control vector $\boldsymbol{u} = [T, \gamma, \beta]$ is composed, respectively, by the thrust, the aerodynamic flight path angle and the speed-brakes deployment.

The dynamics of $\boldsymbol{x}$ are expressed by the following set of ordinary differential equations (ODE), which consider a point-mass representation of the aircraft reduced to the well known "gamma-command" model (i.e. vertical equilibrium is assumed):

$$\frac{\mathrm{d}v}{\mathrm{d}t} = \dot{v} = \frac{T(v,h,\pi) - D(v,h,m,\beta)}{m} - g\sin\gamma \tag{9a}$$

$$\frac{\mathrm{d}h}{\mathrm{d}t} = \dot{h} = v\sin\gamma \tag{9b}$$

$$\frac{\mathrm{d}s}{\mathrm{d}t} = \dot{s} = \sqrt{v^2\cos^2\gamma - W_x(s,h)^2} + W_s(s,h) \tag{9c}$$

$$\frac{\mathrm{d}m}{\mathrm{d}t} = \dot{m} = -f(v,h,\pi) \tag{9d}$$

where π is the throttle; D is the aerodynamic drag; W_x and W_s are, respectively, the cross and along path wind components; g is the local gravity acceleration; and f is the total fuel flow.

From an initial aircraft state ($\boldsymbol{x}_0$), a trajectory prediction algorithm [11] aims at computing future states, based on the aircraft intent inputs, weather forecast data and an aircraft performance model.

Mathematically, the aircraft intent of each flight phase i could be given as a certain control vector closing the two degrees of freedom of Eq. (9), plus an end condition [12]. In practise, however, aircraft are not operated following specific T and γ profiles, and these controls are not known beforehand. Instead, climbs and descents are typically composed by constant Mach (M), calibrated airspeed (CAS) or energy share Factor[2](k) segments performed at maximum climb or idle thrust, respectively; while in the cruise phase aircraft typically fly at constant pressure altitude (h_p) and M. Thus, in a generic formulation, two path constraints close the mathematical problem for each phase defining the aircraft trajectory:

$$\boldsymbol{h_i}\left(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{p}\right) = 0 \tag{10}$$

For instance, the first path constraint of an hypothetical phase could enforce to fly at constant pressure altitude (i.e. $\dot{h}_p = 0$), while the second one could restrict the CAS to be constant (i.e. $\dot{v}_{CAS} = 0$). In such case, the two parameters defining the pressure altitude and CAS values of the concerned phase must be specified to perform the numerical integration. An excellent description of the different path constraints that could be defined in an ATM context is given in [13].

The dynamics of $\boldsymbol{x}$ (9) together with the two path constraints (10) form a system of differential algebraic equations (DAE). Provided that the path constraints are meaningful (i.e. they close the mathematical problem), $\boldsymbol{u}$ can be explicitly determined, reducing the DAE system to an ODE system suitable for numerical integration using standard numerical procedures. The integration is performed until reaching the end condition, which triggers the switch to the following phase.

[2]For an aircraft that is changing both altitude and speed, the energy share factor specifies how much the available thrust is allocated to the vertical and speed evolution.

III. Methodology

In this Section, we describe the methodology followed in this work. In Subsection III-A, we give a general description of the methodology. Then, in Subsections III-B, III-C and III-D, we give more details about the different modules in charge of generating the emergency trajectory.

A. High-Level Methodology

The methodology followed to generate a 4D emergency trajectory is depicted in the diagram of Figure 5. It consists of three main stages, developed by the following modules:

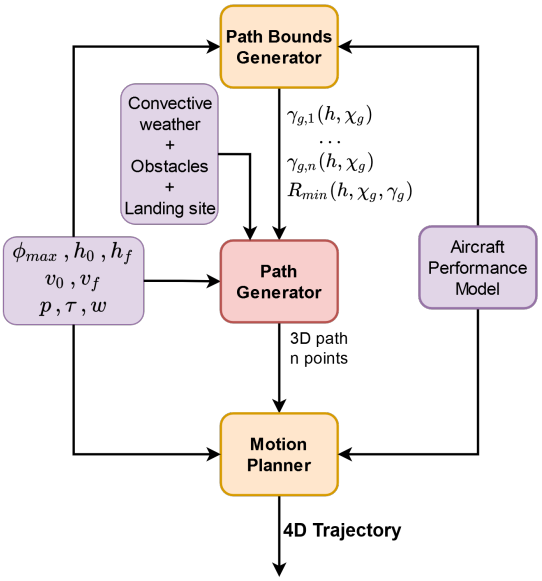


Fig. 5. Generation process of the emergency trajectory

- **Path bounds generator**: in order to define the path linking the aircraft position and the landing site, this module computes a set of flight path angles ($\gamma_{g,1}...\gamma_{g,n}$) and a set of minimum radius of turn ($R_{min,1}...R_{min,n}$) as a function of the altitude and the track (h, χ_g). In order to compute these parameters, both aircraft performance and weather—i.e., temperature, pressure and winds (τ, p and w respectively)—are considered. At this point, a maximum value for the bank angle (which depends on the emergency) is also taken into account (ϕ_{max}). More details regarding the path bounds generator module can be found in Section III-B.

- **Path generator**: the path is generated by taking into account the bounds computed in the previous stage. The method used is based on a combination of the RRT*—used to generate a first version of the path from a set of nodes in a grid—and Dubin curves, used to replace the straight-lines path obtained with RRT* with curves connecting each pair of points. This path links the aircraft position with the landing site, ensuring that all obstacles (and/or convective weather) in the way are avoided. More details regarding the path generator module can be found in Section III-C.
- **Motion planner**: by using the path and the distance to go to the landing site, a 4D trajectory is generated with a TP, which refers to the process of computing a trajectory given a known sequence of control inputs. From an initial state—with associated initial altitude and speed (h_0 and v_0 respectively)—a TP algorithm aims at computing future states, based on the flight intent, weather data and an aircraft performance model (more details in Section III-D).

B. Path Bounds Generator

The path bounds generator, for each of the profiles used by the motion planner (Section III-D), derives a finite number of flight path angles and minimum turn radius as a function of the altitude and the track. Then, these parameters are sent to the path generator. These parameters are computed as follows:

- **Flight path angles (γ_1 to γ_n)**: for any given trajectory, the flight path angle is continuously changing as a function of the altitude (h) and the track (χ)—except if the aircraft follows a fixed-flight-path-angle profile, which usually happens only during the last part of the flight, close to the airport. In order to speed up the computational time of the path generator, the corresponding emergency trajectory is divided in several sections, and a constant flight path angle is derived for each of these sections, as depicted in Figure 6. Thus, the path generator needs to comply with a lower number of flight path angles, which do not continuously change along the trajectory. Still, the path bounds generator computes the flight path angles as a function of h and χ.

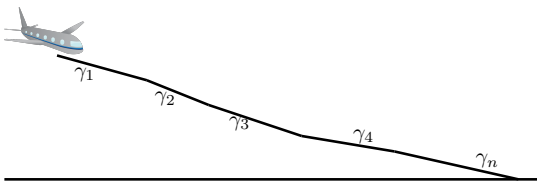


Fig. 6. Example of descent profile with a finite number of flight path angles

- **Minimum turn radius (R_{min})**: it is achieved when the aircraft flies with the maximum bank angle and it is also affected by the aircraft current ground speed. The path bounds generator computes the minimum turn radius as a function of altitude, track, and ground flight path angle (γ_g). A bank angle of 30 degrees is considered if the aircraft is fully maneuverable, while a bank angle of 15 degrees is considered instead.

C. Path Generator

In this section, we describe the path generator, which is the module in charge of generating the path linking the aircraft position and the landing site. It is important to highlight the fact that this path is a three-dimensional path (3D). However, the path generator does not take into account the aircraft performance, so it is the task of the next module, the motion planner (Section III-D), to generate a flyable trajectory by the aircraft. The path computed by the path generator, however, ensures that all the obstacles in the way are avoided. In order to accomplish it, the path generator needs the terrain data, which in this work is represented by a cube. In this cube, each point p generated by the RRT* algorithm is a state vector containing the latitude ψ, the longitude λ, the altitude h and the track χ. In addition, both the track and the altitude of the aircraft are defined within certain bounds:

$$h_{terrain}(\lambda, \psi) < h < h_{max} \tag{11}$$

$$\chi \in [0, 360[\tag{12}$$

where $h_{terrain}$ is the terrain altitude and h_{max} is the aircraft ceiling altitude.

In this study, the path is represented by a set of n Dubins curves $(d_1, d_2, ..., d_n)$ (Figure 7) such as $\forall i, 1 \leq i < n$, the final point of the d_i is the first point of d_{i+1}.

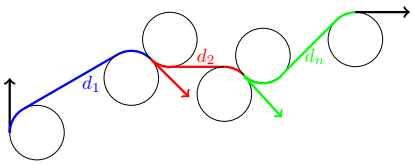


Fig. 7. Path example

The selected objective function is the sum of the distance of each Dubins curve. For a path $(d_1, d_2, ..., d_n)$, the cost is defined as follows :

$$cost(d_1, d_2, ..., d_n) = \sum_{i=1}^{n} distance(d_i) \tag{13}$$

where $distance(d_i)$ is computed as follows :

$$distance(d_i) = \begin{cases} r_1 * \phi_1 + L_S + r_2 * \phi_2 \; if \; CSC \; Dubins \; curve \\ r_1 * \phi_1 + r_2 * \phi_2 + r_3 * \phi_3 \; otherwise \end{cases} \tag{14}$$

One of the main functions of sampling-based path planning algorithms is the free space checking function. The performance of this function is essential to have a very efficient algorithm in terms of computing time. The proposed function only verifies the straight line between the start point and the final point of the Dubins curve. This implies enlarging the obstacles in order to be sure that Dubins curves will be in the

free space. In some cases, the straight line is in the free space but the Dubins curve passes through an obstacle (Figure 8).

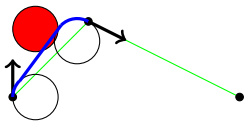


Fig. 8. Example of collision with an obstacle after the addition of Dubins curves : The straight line path is drawn in green and is in the free space. The Dubins path is in blue and passes through the obstacle in red.

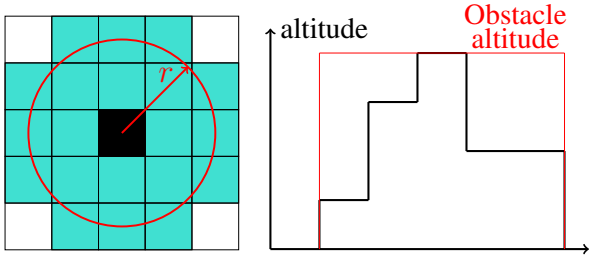
In order to avoid this problem, the obstacles have to be enlarged horizontally by a distance corresponding to the turn radius of the aircraft trajectory. The turn radius, as explained in the previous Subsection (Subsection III-B), depends on the altitude and the track, and the chosen radius is equal to the maximum of all radii. Therefore, the cells located at a horizontal distance lower than this given radius of an obstacle cell become obstacles (Figure 9(a)). Moreover, all cells under these cells are considered also as obstacles (Figure 9(b)).

The path generator, apart from the terrain data, needs to take into account the following constrains:

- **Descent constraints**: the path generator has to take into account the flight path angles computed by the path bounds generator. This means that, in addition to the obstacle constraints, two points can be connected only if the flight path angle between these two points—for the current altitude and track—corresponds to the flight path angle computed with the path bounds generator. The associated constraints can be written as follows (where $\gamma_{profile}$ refers to the flight path angle computed by the path bounds generator):

$$\forall i < n \ \arctan \frac{d_{horizontal}(d_i)}{|z_{end} - z_{start}|} = \gamma_{profile} \qquad (15)$$

- **Track constraints**: in this study, the track constraints taken into account are the following :
  - **Initial track**: at the start position, the aircraft has a given track. The emergency trajectory has to start with this orientation.
  - **Turn radius**: throughout the flight, the pilot changes the track of the aircraft to avoid the obstacles, by following a given turn radius, whose minimum value is computed by the path bounds generator. This radius is used when modeling the turns with Dubins curves.
  - **Landing site track**: the selected landing site has also a given orientation, which means that the final track of the trajectory has to match this orientation.



(a) Horizontal Enlargement (Obstacle cell : Black, New obstacle cells : Blue

(b) Vertical enlargement

Fig. 9. Enlargement of obstacles

The proposed method to generate the path is based on the RRT* algorithm. Each node N generated during the sampling phase is defined by its coordinates in free space. The nodes contain three additional pieces of information to construct the structure of the tree. The first piece of information is the track, which is randomly generated. The second piece of information is the cost of the node N. Finally, the third piece of information is the parent node N_{parent}. It represents the previous node through which the shortest path to reach N passes. At each step of the algorithm, a new random point is generated, located in the free space. Then, this point is steered to its nearest point to be within a distance lower than the neighborhood radius r (Figure 10). Next, the Dubins curve horizontal distance is computed and the altitude of the point $x_{horizontalSteering}$ is modified to have a descent angle which corresponds to the flight path angle generated by the path bounds generator (Figure 11). Finally, the point is connected to the tree if there is no obstacle in the way.

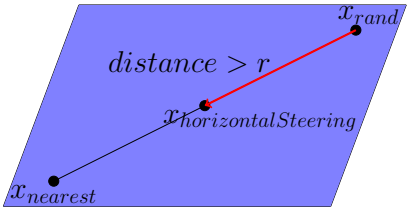


Fig. 10. Horizontal Steering

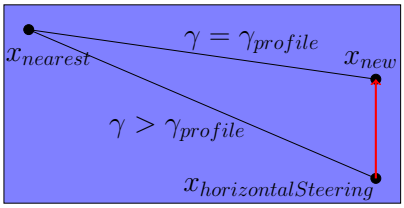


Fig. 11. Vertical Steering

D. Motion Planner

The motion planner is the module in charge of generating a 4D trajectory by using the 3D path obtained from the path generator, together with the current aircraft performance and weather (i.e., temperature, pressure and wind). Different predefined vertical profiles are considered in the motion planner and each one is considered independently. Given a triggering event (i.e. emergency situation for a particular scenario), a profile from this predetermined catalogue is automatically chosen by the tool and the corresponding 4D trajectory is

computed. Furthermore, the path bounds generator (Section III-B) also considers these predefined profiles in order to compute the set of flight path angles that define the profile and the associated minimum turn radius. It is worth noting that these pre-computed profiles are not given in terms of 4D trajectories, but as a sequence of aircraft intents, tailored to each case.

The set of vertical profiles (Table I) considered in this work have been designed to be as generic as possible in order to cover the widest possible range of emergency situations. However, most of emergencies are aircraft model dependent, so the procedure described in each aircraft's flight crew operating manual (FCOM) would ultimately affect the definition of the vertical profiles.

In this work, the TP logic implemented in both the motion planner and the path bounds generator considers the following aspects:

- **Trajectory phases**: the vertical profile of the aircraft trajectory is split in a finite number of phases. Each phase is specified by an end condition and two aircraft intents.
- **Aircraft intents**: operational intents, similar to those one could find in state-of-the-art FMSs, such as flying at constant Mach, CAS, altitude, throttle setting, flight path angle, etc.
- **End condition**: needed to define the transition from one phase to the other where intents might change and/or aerodynamic conditions might change (i.e. usage of flaps, landing gear, etc). Example: a phase is flown at constant Mach and Idle thrust (2 intents) until the moment the CAS achieves a given value (end condition of the phase given in terms of CAS); then, the next phase is flown at constant CAS and constant vertical speed (2 different intents).
- **Trajectory computation (trajectory prediction)**: a full 4D trajectory is unequivocally specified given a sequence of phases, with the 2 intents per phase and the end condition specified. This allows to close the 2 degrees of freedom of the mathematical model describing the movement of the aircraft in the vertical plane (i.e. numerically integrate the equations of movement).

The list of profiles, including how the trajectory is computed in each case are fixed by the motion planner and the pilot will not be able to change them. However, in some cases, more than one outcome could be presented by default to the pilot, who could select the desired trajectory. Moreover, for some phases the pilot could request the system to change the values for (some of) the intents. For instance, in a Mach descent, the pilot could decide the value for Mach, but could not replace that phase for a CAS descent, for instance.

The output of the motion planner component is a 4D trajectory plan. Then, this plan will have to be executed in flight by the (auto) pilot. In nominal operations, the FMS of the aircraft also computes this kind of trajectory plan. A typical example would be the aircraft descent, in where the top of descent (TOD) is fixed. This plan becomes the guidance command for the autopilot, which will try to follow the plan (in managed mode) and adjust the aircraft controls to compensate for uncertainty (mainly due to inaccuracies in the weather forecast and to a lesser extent, inaccuracies in the aircraft performance).

Although the aircraft crew is responsible for deploying speed-brakes, hyper-lift devices (i.e. flaps/slats) and landing gear at the moment they consider the most appropriate, a descent plan computed by a FMS makes some hypotheses on the moments these devices will be deployed. In this work, the same philosophy is followed, taking into account that the descent trajectory is not computed until the runway threshold, but down to a predefined point in space that depends on the type of emergency, health of the aircraft and characteristics of the landing site. Thus, some hypotheses are made regarding the usage of speed-brakes, flap/slats and landing gear, which are specified in each profile definition. The vertical profiles considered in this work depend on the following factors:

- **Engines available**: all engines out or at least one engine is operative.
- **Type of emergency**: Land ASAP (as soon as possible, e.g., fire on cabin) or ANSA (at the nearest suitable airport, e.g. depressurization in cabin).
- **Approach procedure**: airport with or without a suitable IFR (instrumental flight rules) approach procedure.
- **Maneuverability**: aircraft fully or not fully maneuverable.

In addition to these factors, in case no engines are available, the availability of fuel on board is also considered. The combination of these factors, taking into account that some combinations for the "all engines out" cases lead to the same profile, gives as a result the list of profiles of Table I.

TABLE I
LIST OF PROFILES

Profile #	Engines available	Type of emergency	IFR	Maneuverability	Fuel on board
1[3]	No	ASAP	-	-	Yes
2[3]	No	ASAP	-	-	No
3	Yes	ASAP	Yes	Yes	Yes
4	Yes	ASAP	Yes	No	Yes
5	Yes	ASAP	No	Yes	Yes
6	Yes	ASAP	No	No	Yes
7	Yes	ANSA	Yes	Yes	Yes
8	Yes	ANSA	Yes	No	Yes
9	Yes	ANSA	No	Yes	Yes
10	Yes	ANSA	No	No	Yes
11[4]	Yes	ANSA	-	-	Yes

IV. RESULTS

In this Section, we show the results obtained in this work. In Subsection IV-A, we describe the scenario and we explain the inputs needed to apply our methodology. Then, in Subsection

[3] Regardless of the availability of an IFR approach procedure and regardless of the maneuverability status of the aircraft. Note that big aircraft relying on hydraulic power are not fully maneuverable without engine power.

[4] Depressurization or smoke in cabin. For the last part of this emergency trajectory, the aircraft intents from Profiles 7 to 10 are used.

IV-B, we describe the case-studies. Finally, in Subsection IV-C, we show the computed trajectories.

A. Scenario Description and Inputs

The scenario chosen for this work is set in a challenging mountainous area around Grenoble airport, in France, where 15 available landing sites were identified. In order to successfully generate 4D emergency trajectories, the different modules need a series of inputs:

- **Aircraft performance model (APM)**: in this work, we use the base of aircraft data (BADA) APM by Eurocontrol [14], which is a widely and recognised APM in the air traffic management community, providing performance models for a large number of aircraft with very accurate results. Fort all case-studies an Airbus A320-232 has been used.
- **Weather data**: in this work, the longitudinal component of the wind is modelled by a smoothing spline [15], $w : \mathbb{R} \to \mathbb{R}$, such that:

$$w(h) = \sum_{i=1}^{n} c_i B_i(h), \tag{16}$$

  where B_i, $i = 1, \ldots, n_c$ are the B-spline basis functions and $\boldsymbol{c} = [c_1, \ldots, c_n]$ are the control points of the smoothing spline.
  It should be noted that the longitudinal wind has been modelled as a function of the altitude only, as done in similar works [16]. The control points of the spline approximating the longitudinal wind profile are obtained by fitting historical weather data. This data is obtained from gridded binary (GRIB) formatted files provided by the global forecast system (GFS) of the National Oceanic and Atmospheric Administration (NOAA) [17]. Apart from the wind, this data is used as well to obtain the values for temperature and pressure as a function of altitude.
- **Terrain data**: this data is given as a 3D matrix, where altitude data is given every 30 meters. In addition, the location of the landing sites is included in this matrix, each one represented as a 3D point. Furthermore, if the landing site corresponds to an airport, the orientation and the length of the runway is detailed.

B. Case-Studies

Two case-studies have been analyzed in this work. For each case, the corresponding profile from the list of profiles (Table I) has been chosen. In Tables II and III, the different aircraft intents that appear are the following: ALT, maintain a constant altitude; MACH, maintain a constant Mach number; CAS, maintain a constant CAS; ACC: accelerate at a given acceleration or with a given load factor; DEC, decelerate at a given deceleration or with a given load factor; THR: keep a given throttle setting. For the ACC and DEC cases, we have used the energy share factor value proposed by BADA [14], which varies for acceleration and deceleration and depends on the flight phase (descent or climb).

TABLE II
PROFILE 2: ALL ENGINES OUT + ASAP + NO FUEL ON BOARD

Phase	a/c intent #1		a/c intent #2	
Deceleration to green dot[5]	ALT	Current altitude	THR	IDLE
CAS descent	CAS	Green dot	THR	IDLE
Acceleration to increase rate of descent	ACC	-	THR	IDLE
Descent at higher speed to increase rate of descent	CAS	CAS_n	THR	IDLE
Deceleration to V_{app}	DEC	-	THR	IDLE

TABLE III
PROFILE 7: ENGINE(S) OPERATIVE + ANSA + IFR + FULLY MANEUVERABLE

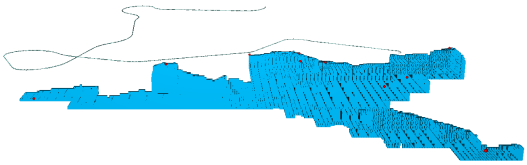
Phase	a/c intent #1		a/c intent #2	
Cruise	MACH	Current Mach	ALT	Current altitude
Mach descent	MACH	Current Mach	THR	IDLE
CAS descent	CAS	CAS_x	THR	IDLE
Deceleration to V_{F1}	DEC	-	THR	IDLE

The two case-studies considered in this work are the following:

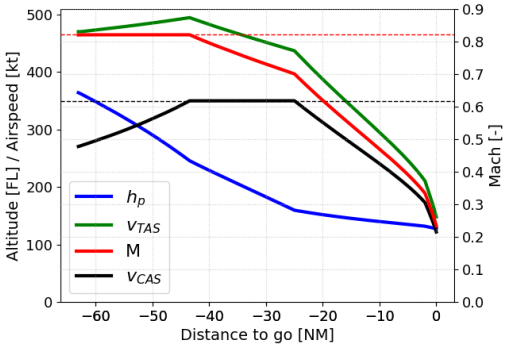
- **Dual-engine failure**: this case-study assumes a dual-engine failure, which leads to a total loss of thrust (as the aircraft considered is a twin-engine aircraft, an A320). The set of phases and intents considered in this case are described in Table II. The aircraft flies at the recommended speed for maximum range until being sure to "make" the best landing site available. Once this is ensured, speed may be increased (to a certain CAS value, CAS_n) to increase rate of descent if necessary. Then, a final deceleration should deliver the aircraft in landing configuration at V_{app}—approach speed, which is the final approach speed when the flaps/slats are in landing configuration and the landing gear is extended—on a 650ft/Nm glide path, and at least 5NM before touchdown area.
- **ANSA case**: this emergency could correspond, for instance, to a fuel leak emergency when the aircraft is in cruise. The set of phases and intents considered in this case are described in Table III. We propose that the aircraft keeps its current (cruise) speed and altitude. Then, a typical Mach/CAS descent (cruise Mach and a certain CAS, CAS_n) is performed until reaching the IAF (initial approach fix), with the aircraft aligned with the initial approach segment and at V_{F1}, which is the target speed for flaps deployed in configuration 1.

For both profiles, the final point is set at a distance of 5NM from the runway, at an altitude of 3250ft with respect to the

[5]For the Airbus A320, the green dot speed is the minimum operating speed in managed mode and clean configuration, being approximately the best lift-to-drag ratio speed.

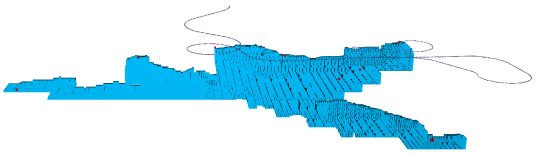


(a) Emergency trajectory and surrounding obstacles

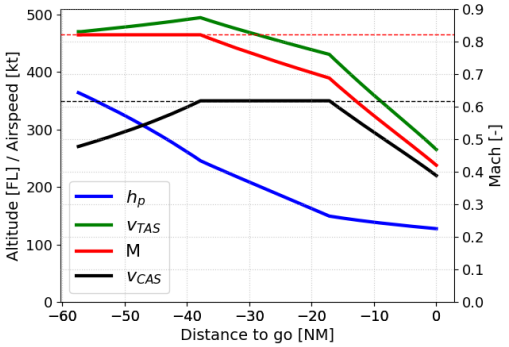


(b) Altitude and speed evolution

Fig. 12. Profile 2



(a) Emergency trajectory and surrounding obstacles



(b) Altitude and speed evolution

Fig. 13. Profile 7

runway altitude[6] and assuming a final aircraft mass of 60,000 kg. In addition, we enforce the aircraft to be aligned with the runway at this point. Finally, the initial altitude is set to 37,000 ft. By taking into account all these constraints, it was impossible to find a feasible trajectory linking the aircraft position with all landing sites. There were cases in which there was an obstacle between the final point and the landing site, making it impossible for the aircraft to reach the landing site by following this type of profiles. In the end, only 3 landing sites in the Grenoble area were reachable. The trajectories depicted in Subsection IV-C were generated for one of these landing sites, located at 9514 ft above mean sea level.

C. Final Trajectories

Figures 12 and 13 show the resulting trajectories for the profiles specified in Section IV-B. In both cases, the first part of both profiles—the cruise phase and the green dot descent in profiles 2 and 7 respectively—is not flown due to the proximity of the landing site. The resulting trajectory follows in both cases an MMO/VMO (maximum operating Mach/maximum operating CAS) profile until the final deceleration to V_{app} and V_{F1} for profiles 2 and 7 respectively.

The computational time for the generation of the trajectory is composed of the computational time spent by the different modules considered in this work. Regarding the path bounds generator, the flight path angles and the minimum turn radius are generated in 7 seconds. A higher computational time is usually spent on the path generator. Depending on the number of iterations, the algorithm computes one, two or three trajectories. Then, in order to ensure that a solution is always found for each airport, the number of iterations needs to be very high (at least 30000). This is due to the fact that the constraints are very restrictive. However, if one solution is sufficient (i.e. one trajectory for one airport), about 10000 iterations are enough. The computational time for the RRT* algorithm increases very fast when the number of iterations increases, leading to a computational time of 3s, 20s, 100s and 500s if the number of iterations is 10000, 30000, 50000, 100000 respectively.

Finally, the motion planner takes between 2 to 5 seconds to generate the 4D trajectory, depending on the profile considered. In general, the emergency trajectories are generated in

[6]For profile 7, as the engines are still working, it might not be needed to finish at such a steep glide path. In this case, the final point could be set at 5NM from the runway and at 1500ft above the runway altitude. Still, in order to simplify the problem, we considered the same final point for both cases.

less than 15 seconds. Still, we are still using a very simple prototype to test our concept; in the future, we expect to optimize our code, as well as to run the tests in a more powerful computer or server.

V. Conclusions

In this work, we showed a novel methodology to generate 4D trajectory plans for emergency situations, designed to safely lead the aircraft to a designated landing site. For that purpose, we used a combination of different methods: the RRT algorithm, Dubins paths and a TP algorithm.

We showed that we are capable of generating emergency trajectories for a wide range of emergency situations in a short amount of time (less than 15 seconds). Ultimately, the concept presented in this paper aims to be part of an in-flight support tool for the FMS. The 4D trajectory plans, after being injected in the FMS, would be executed in-flight by the (auto) pilot.

This work focuses only on the generation of the trajectory; in the future, an insight on the landing site selection process will be given, and a wider range of situations will be investigated in order to assess the feasibility of our concept.

Acknowledgments

We would like to thank Capt. Claude Godel for his valuable advice when defining the vertical profiles used by the motion planner.

References

[1] E. M. Atkins, I. A. Portillo, and M. J. Strube. Emergency Flight Planning Applied to Total Loss of Thrust. Journal of Aircraft, 43(4), 2006.

[2] E. M. Atkins. Emergency landing automation aids: an evaluation inspired by US airways flight 1549. In AIAA Infotech@Aerospace Conference, Atlanta, Georgia, USA, 2010.

[3] P. F. Di Donato and E. M. Atkins. An Off-Runway Emergency Landing Aid for a Small Aircraft Experiencing Loss of Thrust. In AIAA Infotech@Aerospace Conference, Kissimmee, Florida, USA, 2015.

[4] S. Paul, F. Hole, A. Zytek, and C. A. Varela. Flight Trajectory Planning for Fixed-Wing Aircraft in Loss of Thrust Emergencies. In Second International Conference on InfoSymbiotics/DDDAS(Dynamic Data Driven Appliactions Systems), MIT, Cambridge, Massachusetts, USA, 2017.

[5] N. Meuleau, C. Plaunt, E. D. Smith, and T. Smith. An emergency landing planner for damaged aircraft. In 21st Conference on Innovative Applications of Artificial Intelligence, Pasadena, California, USA, 2009.

[6] M. Miwa, T. Takeshi, Y. Satoshi, T. Nobuhiro, and S. Shinji. Real-Time Trajectory Generation Applicable to Emergency Landing Approach. The Japan Society for Aeronautical and Space Sciences, 52(175):21—28, 2009.

[7] Garmin Ltd. AutonomÍ™Autonomous Safety-enhancing Technologies. https://discover.garmin.com/en-US/autonomi/, 2021. Accessed: 2021-04-02.

[8] S. Karaman and E. Frazzoli. Sampling-based Algorithms for Optimal Motion Planning. The International Journal of Robotics Research, 30(7):846—894, 2011.

[9] A. M. Shkel and V. Lumelsky. Classification of the Dubins set. Robotics and Autonomous Systems, 34(4):179—202, 2001.

[10] G. Satyanarayana, D. C. Manyam, A. L. Von Moll, and Z. Fuchs. Shortest Dubins Path to a circle. In AIAA Scitech 2019 Forum, San Diego, California, USA, 2019.

[11] Ramon Dalmau, Marc Melgosa, Santi Vilardaga, and Xavier Prats. A Fast and Flexible Aircraft Trajectory Predictor and Optimiser for ATM Research Applications. In Proceedings of the 8th International Congress on Research in Air Transportation (ICRAT), Castelldefels, Catalonia, Jun 2018. Eurocontrol and FAA.

[12] D. González-Arribas, M. Soler, J. López-Leonés, E. Casado, and Sanjurjo-Rivo M. Automated optimal flight planning based on the aircraft intent description language. Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, 233(3), 2018.

[13] E. Gallo, J. Lopez-Leones, and M. Vilaplana. Trajectory computation infrastructure based on BADA aircraft performance model. In 26th AIAA/IEEE Digital Avionics Systems Conference Proceedings, Dallas, Texas, USA, Jun 2007.

[14] Eurocontrol. User Manual for the Base of Aircraft Data (BADA) Family 4, 2014.

[15] Carl de Boor. On calculating with B-splines. Journal of Approximation Theory, 6(1):50–62, 1972.

[16] Raúl Sáez, Xavier Prats, Tatiana Polishchuk, and Valentin Polishchuk. Traffic Synchronization in Terminal Airspace to Enable Continuous Descent Operations in Trombone Sequencing and Merging Procedures: An Implementation Study for Frankfurt Airport. Transportation Research Part C: Emerging Technologies, 121, 2020.

[17] National Oceanic and Atmospheric Administration (NOAA). Global Forecast System (GFS). https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/global-forcast-system-gfs, 2021. Accessed: 2021-01-22.

Biographies

Mr. **Raúl Sáez** is a Ph.D. candidate in Aerospace Science and Technology at the Technical University of Catalonia (UPC), where he develops his research in the ATM field as part of the ICARUS research group. He received his M.Sc. in Aerospace Engineering from the *Institut Supérieur de l'Aéronautique et de l'Espace* (ISAE Supaero) in 2017 in Toulouse, France, and he obtained his B.Sc. in Aeronautical Engineering from the UPC in 2015.

Ms. **Homeyra Khaledian** is a Ph.D. candidate in Aerospace Science and Technology at the Technical University of Catalonia (UPC). Prior to joining the ICARUS research group, she has worked as a researcher at Sharif University of Technology (2017-2019). She received her M.Sc. in Telecommunication and Signal Processing from Iran University of Science and Technology (IUST) in 2017, and she holds an Electrical engineering degree.

Dr. **Xavier Prats** is a *Serra i Hunter* fellow (associate professor) at UPC. Principal investigator at the ICARUS research group, currently leading the ATM and flight operations research lines. He is an aeronautical engineer from the *École Nationale de l'Aviation Civile* (ENAC), in Toulouse (France); also holds a telecommunications engineering degree from Telecom Barcelona; and received his Ph.D. in Aerospace Science and Technology from UPC.

Mr. **Andréas Guitart** is a Ph.D candidate at *Ecole Nationale de l'Aviation Civile* (ENAC), in Toulouse, France. He received his M.Sc. in Aeronautic Engineering in ENAC in 2019. He is specialized in air traffic systems and optimization.

Pr. **Daniel Delahaye** is a full Professor and Director of the OPTIM research laboratory of ENAC, Toulouse, France. He obtained his Ph.D in automatic control from the Aeronautic and Space National school, Toulouse in 1995 and did a post-doc at the Department of Aeronautics and Astronautics at MIT in 1996.

Pr. **Eric Feron** is a full professor at Georgia Institute of Technology, Atlanta, USA. He obtained his Ph.D. in Aerospace engineering from Stanford University.