DOI: 10.13009/EUCASS2022-4593
9TH EUROPEAN CONFERENCE FOR AERONAUTICS AND SPACE SCIENCES (EUCASS)

Trajectory optimization for the computation of safe emergency trajectories

Maëva Ongale-Obeyi[1⋆], Damien Goubinat[1],
[1]Thales, Toulouse,
first.last@fr.thalesgroup.com
Daniel Delahaye[2], Pierre-Loïc Garoche[2],
[2]ENAC, Toulouse,
first.last@enac.fr

Abstract

Over the past 15 years, considerable progress has been made in building adaptive control systems to help pilots fly damaged aircraft. However, nowadays, in an unexpected event not covered by a procedure, the crew still has to deal with the situation on their own. Indeed they must review the situation on board, then determine the best site for an emergency landing, and finally plan the path to this landing site. In general, the decision depends on many factors, including the actual control envelope of the aircraft, distance to the site, weather, and energy management en route, the approach path's characteristics, the runway or landing site, and emergency facilities at the site. All of these make the crew's workload heavy. So having aid to a decision support module, such as an emergency landing planner, seems to be a must-have.
We, therefore, want to achieve a planner capable of computing a trajectory leading from a known point A to a given point B while avoiding obstacles, a trajectory that would be flyable for a pilot, even without a supporting guidance aid system. So the main characteristics are the fact that the trajectory: is (a) safe, (b) flyable, and (c) following the resources of the aircraft. For this purpose, we combined three algorithms, providing each one of these features: a path-planner, a path smoothing system, and a continuous trajectory system.

I. Introduction

A. State of the art

Among the circumstances that can overload the crew's work, emergencies are the ones that interest us in this article. In response to an emergency during flight, the pilot's first objective is stabilizing and regaining control of the aircraft. Then their focus is diverted to finding the shortest and safest way to land the aircraft, with the assistance of the remaining flight direction system, flight manuals, and air traffic controllers. The decision is difficult because depending on many factors such as the new maneuvering envelope, the rate of available fuel, the distance to available landing fields, terrain avoidance, and minimizing risk to people and property on the ground, among other factors.

Several projects and working groups have often addressed this issue. Different angles of approach have been studied. A recent approach to deal with the case of emergency landings for damaged aircraft can be found in ([1]) but only in considering an obstacle avoidance problem. On a different model, a UAV, it is also an avoidance problem that is dealt with, mainly the management of the landing access during emergency landing ([2]). Another part of the problem is brought in ([3]), where we tackle the generation of a smooth trajectory in order to overcome our problems of guidance aid system failures; in fact, the calculation of a smooth trajectory facilitates gross flight (without assistance) whether for a crew or UAVs.

One lesson from all of his work is that the problem itself is far too complex to tackle all at once, especially in our case, where we try to challenge the computational time. Indeed, either we are talking about avoidance problems as in ([3]), solving problems related to the search for a landing site ([4]), or even trajectory smoothing problems as we can find solutions in ([5]). Also, we notice that, in a general way, the literature presents us with only a partial solution to the problem and not a complete resolution. One of the reasons that can be used as a justification is certification; indeed, by adding the need to have embedded algorithms, we restrain the number of usable systems and, therefore, our way of solving these problems. Thus, by repealing this certification limit, we are aiming for a more comprehensive treatment of the problem. Our ambition is to focus on our resolution system.

B. Our goal

Our goal begins with the problem delimitation; in a desire to simplify our study, we will not focus on the landing site research part. Indeed our problem is limited to calculating one or more trajectories between a starting point and a well-defined arrival point. Using tools from avoidance problems, path smoothing tools, and optimal control theory, we cross these domains to allocate performance to solve the objective on several fronts.

The mathematical formulation of the safety trajectories is related to the avoidance analysis of dynamical systems. This analysis aims to partition the problem into three categories: first, the avoidance problem in a 3D environment. Second, the

DOI: 10.13009/EUCASS2022-4593

study of smoothing tools for paths to have flyable paths without sudden variations of the input parameters, and finally, a study of the situation and the dynamic context in which the vehicle registers. This study makes it possible to satisfy all the criteria that we seek to attribute to our framework, namely:

1) Compute one or more paths in a 3D cluttered environment
2) Compute one or more smooth paths in a 3D cluttered environment
3) Compute one or more continuous and dynamic trajectories in a 3D cluttered environment..

This paper is organized as follows; first we present our theoretical approach, then our numerical experiments.

II. Theoretical approach

A. *Problem Statement*

As we said previously, the problem of emergency trajectories is too dense in its formulation to be treated in its entirety. So, we choose to break down the problem to better manage all aspects. To do this decomposition, we based ourselves on the characteristics that we want our trajectories to have:

1) First, we want to compute a path connecting two points in a known 3D environment. This problem is classic casework widely studied in avoidance problem studies.
2) Then in the case of a problem with the trajectory tracking tools, it is the pilot's responsibility to ensure the continuation of the flight. Thus, it is necessary to produce a trajectory as smooth as possible by avoiding sudden changes in speed or angles. This issue is related to a smoothing path problem.
3) Finally, context forces us to consider a dynamic model. Indeed, taking into account the aircraft's dynamics, the state of the flight controls, and the prediction of the movement of obstacles are essential. This study of dynamic models makes it possible, plus it calculates trajectories in opposition to paths that are a succession of waypoints. So the treatment of this last problem will allow us to pass from paths to trajectories

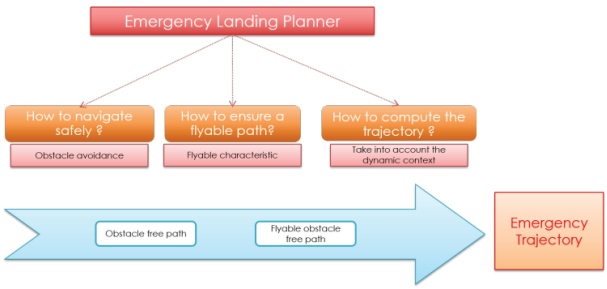


Fig. 1: Our problem division

B. *The avoidance problem : How to navigate safely ?*

Motion determination for an automated robot has been widely discussed over the past fifty years. Two main categories of approaches were usually distinguished: deliberative and reactive (cf Fig.2).

- The principle of deliberative approaches is to determine a complete movement of the robot between an initial position and a final position from a model of the environment in which the system evolves as ultimately as possible.
- Reactive approaches, on the other hand, only calculate the movement to be applied to the next step from sensor data retrieved by the robotic system at each instant. A representation of the environment is thus constructed as the movement progresses: navigation is, therefore, possible in an uncertain environment as well as in a dynamic environment.

DOI: 10.13009/EUCASS2022-4593

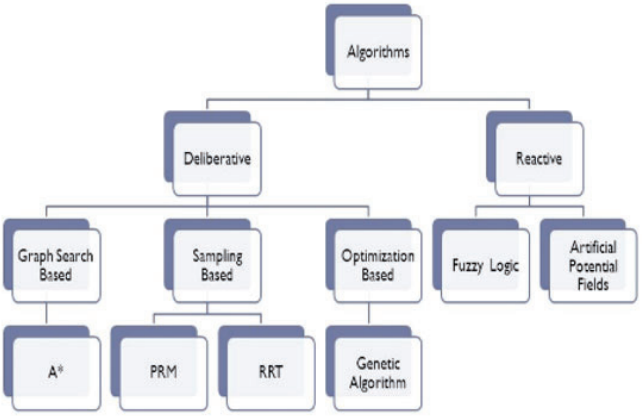


Fig. 2: Classification of path planning algorithms

Because of the importance of having a complete path, deliberative approaches seem best suited to our problem. Thus we have benchmarked some deliberative algorithms to find the one that would best respond to the problem.

The Rapidly exploring Random Trees (RRT) initially presented by Lavalle represents one of the most famous motion planning approaches today. Like randomly sampling the configuration space, RRT starts at the starting location and randomly grows a tree to span the space. The main objective is to favor the extension of the tree to areas that have not yet been filled. The planner pushes the search tree from previously constructed vertices.
In fact, the principle is as follows: from an initial configuration q_{init}, the environment of the robot is explored by arbitrarily choosing a new unobstructed configuration q_{rand}. The second step is to determine the q_{near} node closest to q_{rand} in the existing tree. The next idea would be to try to extend the tree from q_{near} in the direction of q_{rand} by a length ϵ. Finally if this extension succeeds, the newly created configuration q_{new} will be added in the tree. This process will be repeated until the initial q_{goal}vconfiguration is reached. This principle is illustrated in the following figure.

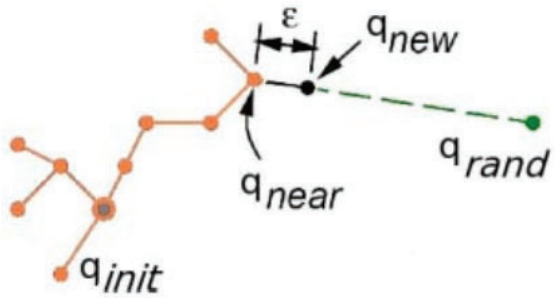


Fig. 3: RRT process

The RRT exploration method consists of a free space spanning tree construction phase and a query phase. The performance of this method comes from the fact that it does not require a pre-calculation phase. An exciting property of this approach is that tree growth is strongly biased towards unexplored areas of configuration space, so rapid exploration occurs. This construction allows, in particular, the processing of large dimensions; thus, the search remains fast in large spaces of immense dimensions. They are also well suited to capture dynamic or non-holonomic constraints and therefore compensate wonderfully for PRM methods that have difficulties in this aspect. Although, at first sight, this planner does not take into account the issue of path cost, there are many versions of the RRT that can do the trick.

We have chosen to focus on RRT (Rapidly Exploring Random Tree) mainly because of its superiority in computing time when it has come to work in high dimensions. Once the free-obstacle path is calculated, we realize that it can present particular angles which are not flyable; thus, a path-smoothing algorithm is needed.

C. The smoothing problem: How can the trajectory be flyable in any case?

In order to prevent any tracking loss issues, we need to be able to provide a smooth path. Also, the need to process the path calculated by our path-planning algorithm comes from this.

DOI: 10.13009/EUCASS2022-4593

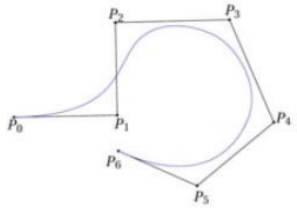
Planning paths for a non-holonomic mobile robot is a concern of many works. The first consist of generating paths made up of straight-line segments tangentially connecting arcs of circles of maximum curvature. These paths are the shortest in configurations where the robot only moves forward (Dubins demonstrated this in 1957). However, several studies on the control of mobile robots have highlighted the importance of continuity of curvature to obtain paths whose tracking is precise. However, the paths already mentioned do not verify this property. Therefore, solutions have been proposed by introducing interpolation and approximation techniques to geometrically model paths for mobile robots. Among these primitives, we studied cubic splines, the first methods to be developed. Then Bezier curves were introduced, adopting a different and more flexible design. The evolution continued with B-Splines, the generalization of Bezier's curves, and Non-Uniform Rational Base-Splines (NURBS) in the 1980s. This latter has proved to have significant assets, so we focus on this method.

A NURBS curve of degree p is defined by :

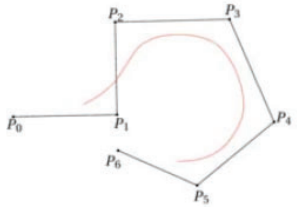
$$C(u) = \frac{\sum_{i=0}^{n} N_{i,p}(u) w_i P_i}{\sum_{i=0}^{n} N_{i,p}(u) w_i} \quad a \leq u \leq b \tag{1}$$

Where the $\{P_i\}$ are the control points (forming a control polygon) and the $\{w_i\}$ are their associated weights. The $\{N_{i,p}(u)\}$ are the B-splinar basis functions of degree p defined on the nodal vector U.. The nodal vector is a sequence of parameter values that determine where and how the control points will affect the shape of the curve. It divides the parameter space into intervals, and each time the parameter enters a new nodal interval, a new control point becomes active while an older one becomes inactive. This, therefore, guarantees the local influence of the control points. This vector is denied by an increasing sequence of nodes $(u_i \leq u_{i+1})$ between 0 and 1. These consecutive nodes can have identical values; this is defined by the multiplicity of a node, which makes it possible to accentuate the influence of a point on the curve.

Thus, to ensure the interpolation of the two ends of the control polygon, it is necessary to fix the multiplicity of the first and the last node of U to the value $p+1$ (Fig. 4). Note also that the degree p , the number of control points $n+1$ and the number of nodes $m+1$ of the nodal vector are related by: $m = n + p + 1$.



(a) $U = (0, 0, 0, 0, \frac{1}{4}, \frac{1}{2}, \frac{3}{4}, 1, 1, 1, 1)$.



(b) $U = (0, \frac{1}{10}, \frac{1}{5}, \frac{3}{10}, \frac{2}{5}, \frac{1}{2}, \frac{3}{5}, \frac{7}{10}, \frac{4}{5}, \frac{9}{10}, 1)$.

Fig. 4: NURBS curves with the same control polygon and different nodal vectors [5].

As for the weight parameter, each w_i determines the influence of the point P_i on the curve. Increasing the weight associated with a point pulls the curve towards that point, whereas when the weight is decreased, the curve moves away from the point (Figure 5).

If $w_i = 1$ for any point, we find ourselves in the case of B-splines. Otherwise, for values different from 1, greater or lesser importance is attributed to the corresponding control point :

- $w_i \leq 1$: gives a curve less close to P_i

DOI: 10.13009/EUCASS2022-4593

- $w_i \geq 1$: gives a curve closer to P_i.
- $w_i = 0$: the point P_i no longer has any influence.
- $w_i \to \infty$: the curve passes through P_i.

NURBS are B-splines with weighted control points, allowing the curve to be snapped to one or more points as desired. They maintain the degree of independence and the property of local modification that characterize B-splines. Incorporating control point weights improves flexibility and allows NURBS to synthesize different curve shapes, changing control points, nodes and weights.

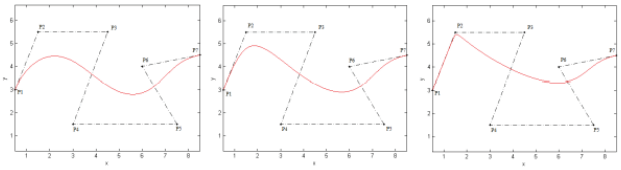


Fig. 5: NURBS curve with variation of the weight of P_2.

NURBS has proved to be the most advantageous due to its attractive properties such as local control and high rate of flexibility. This rate is explained by the possibility of acting on the curve to have the desired shape by modifying either the nodal vector, the positions of the control points, or the weights associated with these points. This allows for smooth and ergonomic curves without forgetting the criterion of the simplicity of construction and implementation and the low complexity of the algorithms used.

D. The continuous problem: How to plan the trajectory?

Now that the paths have been smoothed out, we are using these waypoints as a landmark to calculate continuous paths avoiding obstacles.

This paper uses the Point Mass Model for general flight vehicle dynamics. Consider an inertial frame attached to the Earth. The state of the system is defined by :

- $(x, y, z) \in \mathbb{R}^3$ is the position of a vehicule in the inertial reference frame.
- $V \in [V_{min}, V_{max}]$ is the True Air Speed
- $\psi \in [-\pi, \pi]$ is the heading angle
- $\gamma \in [-\pi, \pi]$ is the flight pass angle

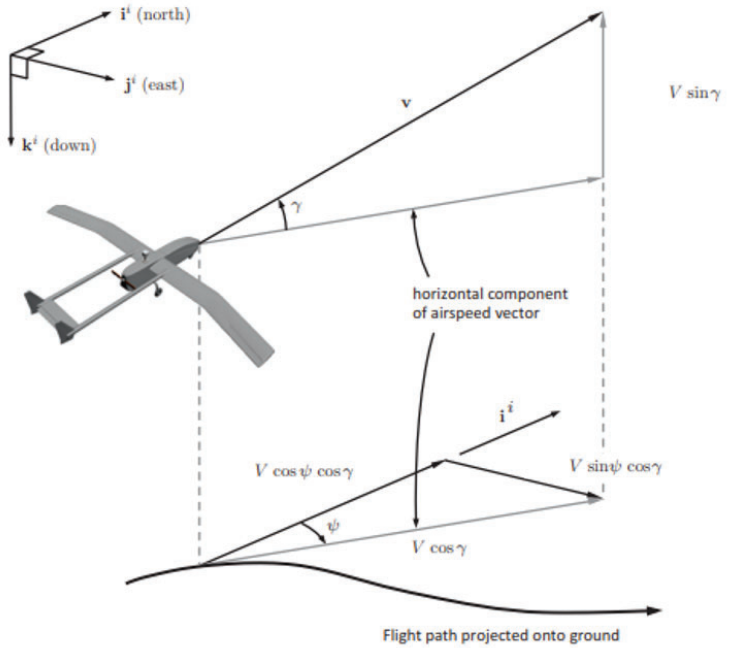


Fig. 6: Different angles of our air vehicles

DOI: 10.13009/EUCASS2022-4593

The calculation of continuous trajectory is based on a modeling of the equations of motion, which are defined by the following system as follows:

$$\left\{\begin{array}{lcl} \dot{x} & = & V\sin\psi\cos\gamma \\ \dot{y} & = & V\cos\psi\cos\gamma \\ \dot{z} & = & V\sin\gamma \\ \dot{\psi} & = & u_1 \\ \dot{\gamma} & = & u_2\ , \end{array}\right. \tag{2}$$

where $V(t) = \|v(t)\|$, $v(t) = (\dot{x}, \dot{y}, \dot{z}) \in \mathbb{R}^3$ is the velocity at the time t, ψ is the heading angle and γ the flight-path angle. Here the control $u = (u_1, u_2)$ is submitted to the constraint

$$\|u_1\| \leq u_{max}\ ,\ \|u_2\| \leq u_{max}, \tag{3}$$

where the typical value of the maximal control u_{max} is arond 1.

Based on this system of equations, we seek to calculate a trajectory that would minimize a specific criterion while respecting the present constraints. The resolution of this type of problem calls upon principles of the theory of optimal control, which we are going to state the theorem pillar:

Definition 2.1: In optimal control, we distinguish two vector variables: a control variable u, to indicate the decisions made, and a variable x, to indicate the state of the system over time. The standard control model (P_c) for a control system is composed of two essential elements: the dynamic controlled system (to which constraints are often added) and the functional (defining the optimization criterion):

$$(P_c) = \left\{\begin{array}{l} \underset{u,x}{min}\mathcal{J}(u(t), x(t)),\ \text{functional corresponding to a minimizing cost.} \\ \text{under the constraints} \\ \dot{x} = f(u(t), x(t), t),\ \text{controlled dynamic system: equations of state} \\ x(t_0) = x^{(t_0)}, x(t_f) \in \mathcal{C}_f,\ \text{initial and final conditions, on the states} \\ g(u(t), x(t), t) \leq 0,\ \text{any conditions on the control and/or the states} \end{array}\right. \tag{4}$$

Theorem 2.1: If the trajectory $x(\cdot)$, associated to the optimal control u on $[t_0, t_f]$, is optimal, then it is the projection of an extremal $(x(\cdot), p(\cdot), p^0, u(\cdot))$ (called extremal lift), where $p^0 \neq 0$ and $p(\cdot) : [t_0, t_f] \to \mathbb{R}^n$ is an absolutely continuous mapping called adjoint vector, with $(p(\cdot), p^0) \neq (0, 0)$, such that

$$\left\{\begin{array}{l} \dot{x}(t) = \dfrac{\partial H}{\partial p}(x(t), p(t), p^0, u(t), t) \\ \dot{p}(t) = -\dfrac{\partial H}{\partial x}(x(t), p(t), p^0, u(t), t) \end{array}\right. \tag{5}$$

almost everywhere on $[t_0, t_f]$, where $H(x, p, p^0, u, t) = \langle p, f(x, u, t)\rangle + p^0 f^0(x, u, t)$ is the Hamiltonian, and there holds

$$H(x(t), p(t), p^0, u(t), t) = \underset{v\in\mathcal{U}}{max}\ H(x(t), p(t), p^0, v(t), t) \tag{6}$$

almost everywhere on $[t_0, t_f]$. If moreover the final time t_f to reach the target M_1 is not fixed, the one has the condition at the final time t_f

$$\underset{v\in\mathcal{U}}{max}\ H(x(t), p(t), p^0, v(t), t) = -p^0\frac{\partial g}{\partial t}(x(t_f), t_f). \tag{7}$$

Additionally, if M_0 and M_1 (or just one of them) are submanifolds of $\mathbb{R}^n$ locally around $x(t_0) \in M_0$ and $x(t_f) \in M_1$, then the adjoint vector can be built in order to satisfy the transversality conditions at both extremities (or just one of them)

$$p(t_0) \perp T_{x(t_0)}M_0\ ,\ p(t_f) - p^0\frac{\partial g}{\partial t}(x(t_f), t_f) \perp T_{x(t_f)}M_1, \tag{8}$$

where T_xM_i denotes the tangent space to M_i at the point x_i.

Many numerical methods for optimal control have been developed in recent decades. Without making an exhaustive list of all the methods, we will focus on the local deterministic methods, which are distinguished into two subsets:

DOI: 10.13009/EUCASS2022-4593

- Indirect methods: These indirect methods use the conditions of the PMP to determine the controls u as a function of the states $x$and of the conjoint states p, and reduce to the resolution of an algebra-differential system. This last system will be transformed into a nonlinear problem of finite dimension, in which it is necessary to determine the initial assistant states $p(t_0)$ making it possible to obtain the final states $x(t_f)$.
- Direct methods: By appealing to a total or partial "discretization" of the optimal control problem (in the sense that discretization makes it possible to transform the continuous problem into a discrete problem with a finite number of variables), they refer partly to the original problem into a (significant) nonlinear programming problem.

However, the literature has allowed us to identify the advantages and disadvantages of these methods, which we have summarized in the table below and which we can find in ([6]).

Direct Methods	Indirect Methods
Simple implementation, without prior knowledge	A priori knowledge of the structure of the optimal trajectory
Insensitive to choice of inital condition	Very sensitive ti the choice of the initial condition
Ease of taking state constraints into account	Theoretical difficulty of taking state constraints into account
Globally optimal closed-loop controls	Locally optimal open-loop controls
Low and medium numerical precision	Very high numerical precision
Effective in low dimension	Effective in any dimension
Problem of local minima	Small domain of convergence
Heavy in memory	Parallelizable calculations

When we step back from our problem, we have instead leaned towards direct methods. Among the direct method, collocation is the most efficient in our case ([7]). Already used to compute satellite trajectories, the collocation method is now accurate and robust.

The general principle is the following. We assume a specific representation of the state, "polynomial," in particular; then, one forces the satisfaction of the dynamic equations (in the sense that the controlled dynamic system is verified) in a finite number of instants under the collocation conditions. These intermediate instants in $[t_l, t_{l+1}[$, called quadrature instants, subdivide $[t_l, t_{l+1}[$ into $k-1$ periods, and are used by the digital dynamics integration scheme. Piecewise polynomial state trajectories are obtained, which converge towards the dynamics of the controlled system, at the collocation points. The discretization of the problem in optimal control is then carried out and also generates an NLP problem.

A disadvantage of direct fire methods is the dependence on the initial point. Indeed, it is common for the digital integration process to be subject to numerical instabilities. For example, small variations on the initial conditions x_0 can lead, in certain cases, to a strong variation on the terminal conditions. Thus the collocation method seems to be the one that would be the most suitable for our problem; but again we will judge this empirically.

III. Numerical experiments

We have tested our initial prototype planner in a standalone way on two different scenarios generated using accurate terrain data. In both scenarios, we are working with solid constraints, namely :

- the obstacles database is reduced to terrain data. So we only work on static obstacles and will neglect the effects of the wind;
- whether working on UAVs or aircraft, the speeds of the carriers are constant;
- in both cases, we work in a 3D environment;
- our trajectories must minimize the time criterion

A. UAVs case

The considered configuration is reported in Fig.7 .

DOI: 10.13009/EUCASS2022-4593

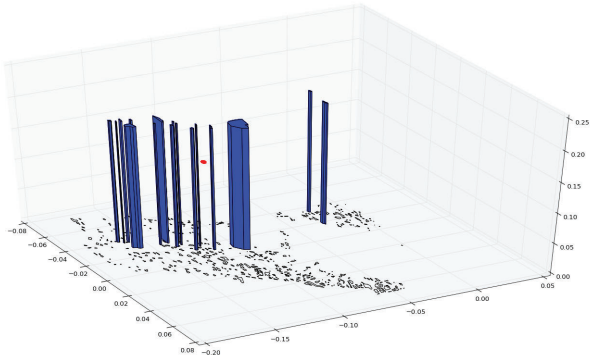


Fig. 7: UAVs environment : The starting positions are marked with a red point, and the destination positions are black shapes. The blue columns represent the no-fly zones, obstacles that must be avoided.

First, three paths are calculated in the previously defined environment (see Fig.8). Starting from the same starting point, they connect different landing sites chosen randomly by the algorithm.

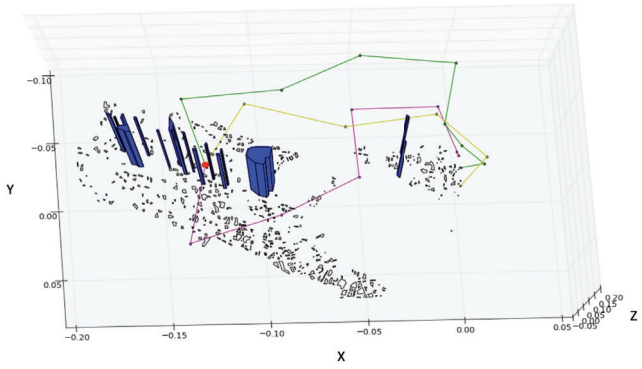


Fig. 8: Computation of the flight plans: For visibility, the waypoints are linked together; in reality, they are not.

Then, we smooth them out using the previously calculated path, Fig.9.

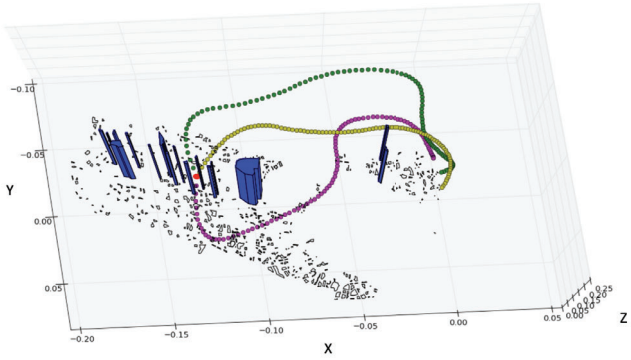


Fig. 9: Computation of the smooth flight plans.

Finally, now that the flight plans are well established, we use them as a reference to calculate the continuous trajectories. Considering the UAV's dynamics, previously defined in (2), we can calculate three 5D trajectories avoiding obstacles, Fig. 10.

DOI: 10.13009/EUCASS2022-4593

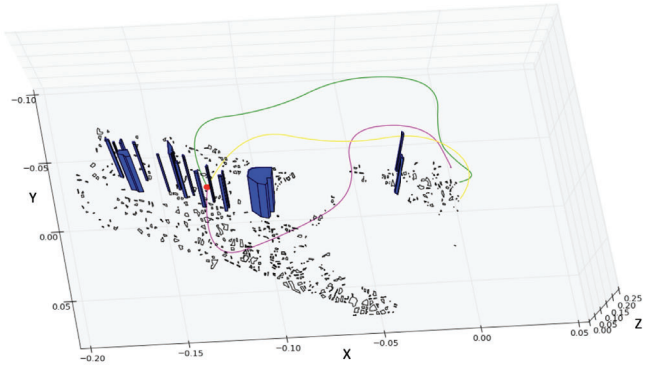


Fig. 10: Computation of the dynamic trajectories.

B. Aircraft case

Unlike the previous scenario, this environment has only one landing site. Then, the algorithm will calculate trajectories to the same destination. In our application, the obstacles database is only composed of the terrain relief (Fig.11). The input of considerable size strongly affects the calculation times, which are three times higher than those of the previous scenario.

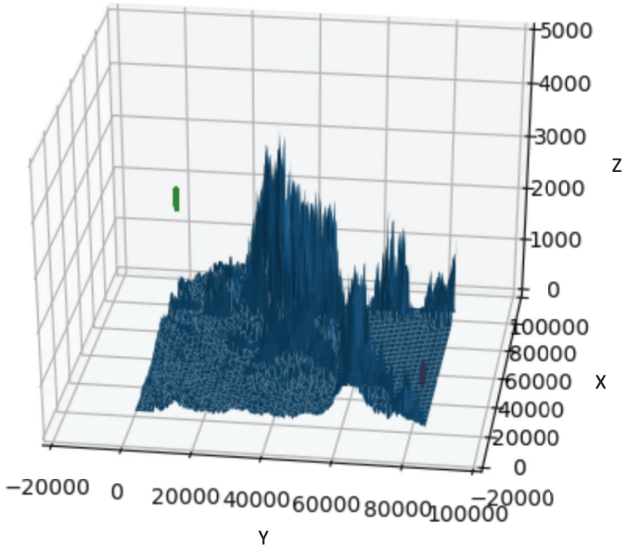


Fig. 11: Environment: the terrain is represented in blue, the red dot symbolizes the landing site position the green dot identifies the starting position.

Traditionally, we know that an airplane follows a particular flight pattern; however, in our case, we are working in an emergency. Thus we considered for the moment that we could abrogate flight phase constraints. Compared to the first scenario, the size of the terrain database is too essential to generate a large number of trajectories, so we will try to calculate only two trajectories.

DOI: 10.13009/EUCASS2022-4593

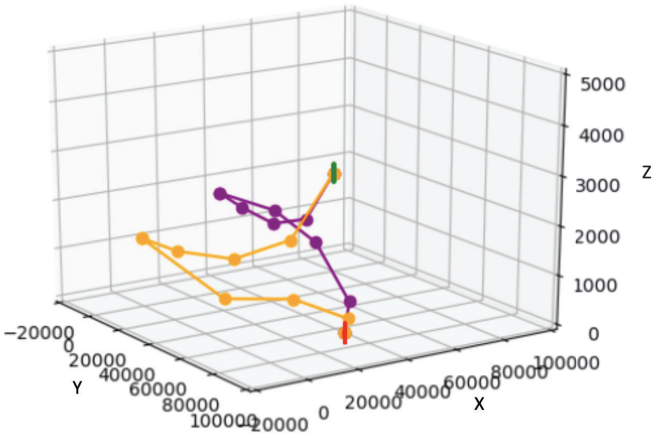


Fig. 12: This part is only dedicated to obstacle avoidance: we work on the paths traveling in the obstacles.

The two paths still present too many non-flyable angles, and the smoothing part of the framework is activated.

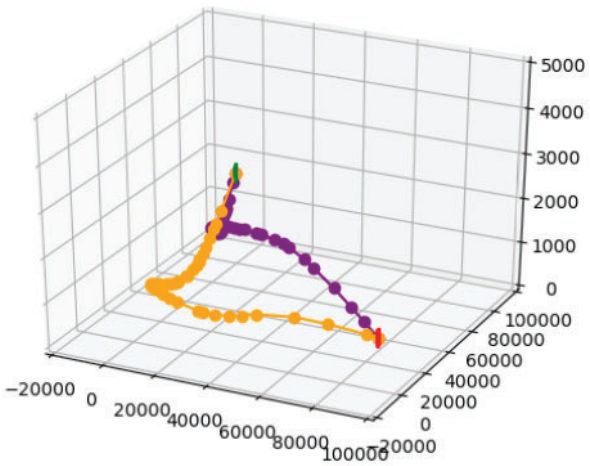
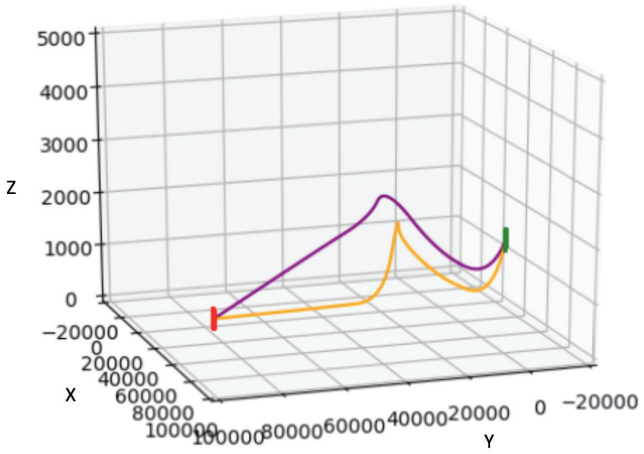


Fig. 13: The undesirable angles have been erased by the smoothing, and only the consideration of the dynamics of the aircraft is missing.



DOI: 10.13009/EUCASS2022-4593

Fig. 14: Finally, the emergency trajectories are calculated after passing through the continuous trajectory calculation module.

Regarding the time, we are of the order of 10 s when we dwell on the computation of three trajectories in the case of a UAV; in this context, this calculation time is considered reasonable. On the other hand, for the avionics case, the calculation of only two trajectories results in 4 to 5 minutes. This was foreseeable given the size of the obstacle database (3907 Ko for the avionics one against 4 Ko for the UAVs one). However, this time remains unacceptable given the emergency in which we are involved.

IV. Conclusion

This thesis contributes to the architecture of a complete system, including both path planning and trajectories planning algorithms in case of emergency, see figure below. It is mandatory for the planner to quickly compute several trajectories towards different known sites, taking the environment and the UAV's motion into account. The computation combines a RRT and then smoothens the computed solutions with NURBS methods to obtain a set of smooth paths in a 3D cluttered environment. This path planning part produces many reference paths that are supplied to the second part of our planner. Based on a collocation method, this part is dedicated to the trajectories planning. The approach uses a motion equation system based on 6D Dubin's dynamics of the vehicle in order to output the expected trajectories.

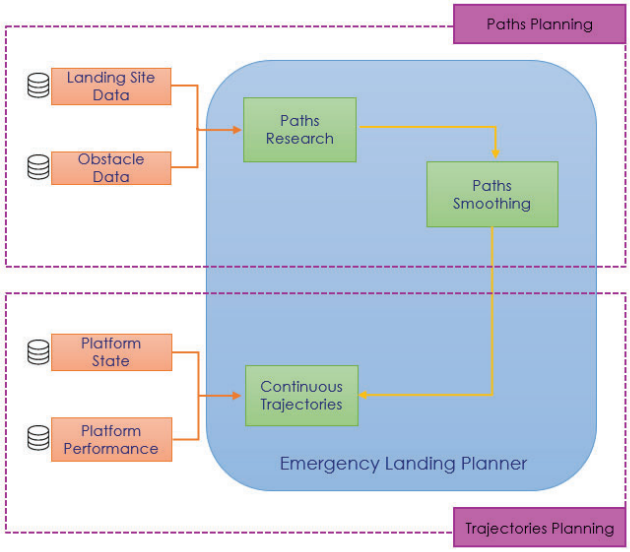


Fig. 15: Our architecture

Future work includes the implementation of MultiTrajectories in a 3D dynamic environment, using wind data and no constant speed in the dynamics. Also, the approach could be extended to large-scale multi-path systems by appropriately parallelizing the computation to reduce computation time. This requires further investigations. Finally, and above all, the question of real-time computation is a significant problem to address and which it would be interesting to look into.

References

[1] , *An emergency landing planner for damaged aircraft*, Meuleau, Nicolas F and Plaunt, Christian J and Smith, David E and Smith, Tristan B, Twenty-first IAAI conference, 2009

[2] , *Any-time trajectory planning for safe emergency landing*, Váňa, Petr and Sláma, Jakub and Faigl, Jan and Pačes, Pavel, 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 5691–5696, 2018, IEEE

[3] , *Modified Dubins parameterization for aircraft emergency trajectory planning*, Yomchinda, Thanan and Horn, Joseph F and Langelaan, Jack W, Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering, 231, 2, 374–393, 2017, SAGE Publications Sage UK: London, England

[4] , *Planning for landing site selection in the aerial supply delivery*, Kushleyev, Aleksandr and MacAllister, Brian and Likhachev, Maxim, 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, 1146–1153, 2011, IEEE

[5] , *Optimisation de la navigation robotique*, Jalel, Sawssen, 2016

[6] , *Contrôle optimal: théorie & applications*, Trélat, Emmanuel, 36, 2005, Vuibert Paris

[7] , *Sur la résolution numérique de problèmes de contrôle optimal à solution bang-bang via les méthodes homotopiques*, Gergaud, J, Hdr, Institut National Polytechnique de Toulouse, Toulouse, France, 2008