Aircraft cruise alternative trajectories generation: a mixed RRG-clustering approach

Jean-Claude Lebègue, Andreas Guitart, Céline Demouge, Daniel Delahaye, Jacco Hoekstra, Eric Feron

Aircraft cruise alternative trajectories generation: a mixed RRG-clustering approach

Jean-Claude Lebegue[1,2][0009−0005−8208−2496], Andréas Guitart[1][0000−0002−1944−2250], Céline Demouge[1][0000−0002−6024−2840], Daniel Delahaye[1], Jacco Hoekstra[3], and Eric Feron[4]

[1] Ecole Nationale de L'Aviation Civile, Toulouse, France
firstname.lastname@enac.fr
[2] Sopra Steria, Toulouse, France jean-claude.lebegue@soprasteria.com
[3] TU Delft, Delft, Netherlands J.M.Hoekstra@tudelft.nl
[4] King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
eric.feron@kaust.edu.sa

Abstract. Weather obstacles in the airspace can interfere with an aircraft's flight plan. Pilots, assisted by air traffic controllers (ATCs), perform avoidance maneuvers that can be optimized. This paper addresses the generation of alternative aircraft trajectories to resolve unexpected events. The authors propose a solution based on the RRG algorithm, K-means clustering, and Dynamic Time Warping (DTW) similarity metric to address the problem. The mixed algorithm succeeds in generating a set of paths with diversity in an obstacle constrained airspace between Paris-Toulouse and London-Toulouse airports. This tool could help to reduce the workload of pilots and ATCs when such a situation arises.

Keywords: metrics · RRG · alternative trajectory · clustering · similarity.

1 Introduction

Aircraft trajectory optimization is one of the most important topics within the frame of air transport. To minimize fuel consumption, each airline computes an optimized flight plan. This optimization takes into account the aircraft's performance and the expected weather. This flight plan is followed as much as possible by the pilots. However, during a flight, an aircraft can be confronted with unexpected events that can disrupt the flight plan: weather obstacles, conflicts between aircraft, or "mechanical" failures. In the last case, pilots often have to land as soon as possible. This problem has been addressed in the European project SafeNcy [6, 25, 26, 30]. In the two other cases, pilots led by air traffic controllers have to find solutions to avoid hazardous areas or to solve conflicts. The resolution should be done fast, that is why these solutions are solely optimal. Moreover, in some very critical situations, it could be difficult to rapidly react and automation could help. We propose to address this problem by automatically generating, before the flight, alternative cruise trajectories to rapidly have a satisfactory solution.

Moreover, air transport is facing a new major challenge: reducing its environmental impact. Indeed, aviation is responsible for about 3-5% of total global warming [20]. Its impact is composed of CO_2 but also from non-CO_2 effects. Among these non-CO_2 effects, condensation trails (contrails) have the higher warming power [20]. Indeed, if they persist and form cirrus clouds, contrails can have a warming effect. Their impact is still not well known and contrail areas are still difficult to predict perfectly [10] even if ice-supersaturated areas are the most likely to be favorable to persistent contrails [14]. These areas can therefore be considered as hard obstacles such as thunderstorm areas, or be considered as soft obstacles areas where the crossing time must be limited. Contrails seem to have similarities with thunderstorm areas, that is why, the generation of alternative trajectories could be a solution to their avoidance.

Most of the works in the literature focus on the generation of one avoidance trajectory for a given situation. In these works, authors propose to cure the situation by computing one avoidance trajectory at the occurrence of the obstacles. In contrast, in this paper, we propose to generate several trajectories to prevent an unexpected event.

The objective of this research is to create a solution framework to design, for a given flight, efficient alternative cruise trajectories to ensure obstacle avoidance. Using the alternative trajectories, the aircraft will be able to deviate from its initial trajectory and join another trajectory. It can then resume the initial flight plan or continue to follow the new route. In this paper, the following requirements are considered:

- Alternative trajectories should not be too far (fuel is limited) to the optimal flight plan or too similar to each other. To avoid this, a similarity metric is defined.
- An alternative trajectory has at least one significant maneuver compared to the initial flight plan.
- At least one of the alternative paths has to avoid the obstacles.

The paper is organized as follows: Section 2 presents a state of the art about alternative trajectories generation and similarity metrics. Then, Section 3 details the proposed algorithm. Finally, in Section 4, the test results on contrails avoidance are presented.

2 Prior works

This section presents prior works related to alternative path algorithms and some similarity metrics.

2.1 Path Generation algorithms

Many approaches have been developed to find a solution to the path planning problem. Among them, optimal control [4, 16, 24], graph [2, 21], or front propagation [22, 27, 28] approaches can be mentioned. This state-of-the-art focuses

on sampling-based path planning methods. This type of methods has been first introduced by LaValle [19]. In his paper, he presents a new algorithm called Rapidly-exploring Random Tree (RRT). This algorithm generates a tree in order to find a path. The tree grows at each iteration by adding a new point. An extension of this method for graph generation called RRG has been developed by Karaman and Frazzoli [17]. It will be detailed in Section 3. More recently, Janson, Schmerling, Clark, and Pavone [15] proposed a new sampling-based path planning algorithm called Fast Marching Tree Star (FMT*). This method performs a forward propagation over several sampled points and creates a tree of paths. FMT* is asymptotically optimal and faster than the RRT algorithm. Initially used in robotics, these methods have recently also been used in aviation [3, 9, 11, 12, 26].

2.2 Alternative path algorithms

The alternative path problem is a well known graph problem in the literature, often referred to as the k-shortest path problem. Two main approaches have been used to tackle this problem: graph structure change [32] and ripple spreading [13]. The former approach starts by computing the shortest path between an Origin-Destination (OD) pair thanks to a shortest path algorithm (Dijkstra algorithm [7]). Then, each link of this shortest path is disconnected from the graph one at a time. At each disconnection, the shortest path is recomputed. This path is a potential k-shortest path. When all links have been disconnected and the potential shortest paths computed, the k-shortest paths are determined by sorting and selecting the k least cost paths from this set. Elsewhere, the latter approach generates k ripples from the origin node. When a ripple meets a node, it triggers a new ripple at this node. The process keeps going on until k ripples have reached the destination.

2.3 Similarity metrics

The k-shortest paths achieved by one of the previous methods are often very similar. The number of overlapped links is important. A disruption on a shared link may impact several paths. This is the reason why attention was paid on the k dissimilar shortest path problem [1]. Several different similarity indices were developed to compare paths. Chondrogiannis, Bouros, Gamper, and Leser [5] use the overlap ratio as a measure of similarity. Considering two paths, it is the sum of the weights of the shared links over the length of one path. In [23], the authors define a lower bound length for two paths and a set of heuristics to avoid exploring some unnecessary paths to compute the k shortest paths with diversity. They use a wide variety of path similarity metrics from the literature (see Table 1 where $\ell(p_i)$ is the length of the path p_i and $p_i \cap p_j$ represents the overlap of the paths p_i and p_j). Finally, Talarico, Sörensen, and Springael [31] define a similarity metric to compare two solutions of the Vehicle Routing Problem (VRP). Their metric is similar to $sim_2(p_i, p_j)$ on the second row of Table 1, the difference

is that the similarity of two VRP solutions is the maximum similarity between their routes.

Notation	Definition
$sim_1(p_i, p_j)$	$\frac{\ell(p_i \cap p_j)}{\ell(p_i \cup p_j)}$
$sim_2(p_i, p_j)$	$\frac{\ell(p_i \cap p_j)}{2\ell(p_i)} + \frac{\ell(p_i \cap p_j)}{2\ell(p_j)}$
$sim_3(p_i, p_j)$	$\sqrt{\frac{\ell(p_i \cap p_j)^2}{\ell(p_i)\ell(p_j)}}$
$sim_4(p_i, p_j)$	$\frac{\ell(p_i \cap p_j)}{\max(\ell(p_i), \ell(p_j))}$
$sim_5(p_i, p_j)$	$\frac{\ell(p_i \cap p_j)}{\min(\ell(p_i), \ell(p_j))}$

Table 1: Path similarity metrics from the literature

3 Alternative Path Generation

As presented in the introduction, the objective of this research is to design alternative trajectories that are the following characteristics:

- in all types of situations, at least one trajectory avoids the obstacles;
- two trajectories have at least one different significant maneuver;
- alternative trajectories have all a significant difference with the flight plan but are not too far from it.

To answer the two first issues, it is proposed to use the Rapidly-Exploring Random Graph (RRG) algorithm that can generate several paths between two points. Moreover, it is easily adaptable to consider the constraints of the problem. The third issue is solved by using clusterization and removing similar routes according to a metric.

3.1 Graph generation

In the following paragraphs, a graph G will be represented by a set of nodes V and a set of edges E. Before presenting the details of the RRG algorithm, it is necessary to introduce its main functions. This algorithm uses 4 functions:

- The *sampling* function randomly or uniformly generates sample points in the space.
- The *steer* function brings a random point closer to the graph.
- The *nearest* function returns, for a given point x, the closest point in the graph.
- The *near* function returns, for a given point x, a set of nodes at distance lower than a radius r.

RRG algorithm was first introduced by Karaman and Frazzoli [17, 18] and works in the same way as RRT. Figure 1 shows an iteration of the RRG algorithm. x_{new} is added to the vertex set V and a link is created between the new point x_{new} and its nearest neighbor $x_{nearest}$ as for the RRT algorithm. Then, new connections are added to the edges set E between x_{new} and all vertices x_{near} in V at a distance lower than a radius r depending on the cardinal of V. This radius is defined as follows:

$$r = \min\{\eta, \gamma_{RRG}(\log(|V|)/|V|)^{1/d}\}, \tag{1}$$

where:

$$\gamma_{RRG} > 2(1 + 1/d)^{1/d}(\mu(\chi_{free})/\zeta_d)^{1/d}, \tag{2}$$

with d the dimension of the space ($d = 2$ in this study), $\mu(\chi_{free})$ is the Lebesgue measure [5] of the obstacle-free space and ζ_d is the volume of the unit ball in the d-dimensional euclidean [6].

As for the RRT* algorithm, it is proven to be asymptotically optimal.

Improvements In this study, it is proposed to adapt the RRG algorithm to the alternative trajectories generation.

To avoid too short maneuvers, it is proposed to modify the *near* function. As a reminder, this function returns the closest neighbors. If a neighbor is very close to the new point, it creates a very short connection. To avoid this phenomenon, the points at a distance lower than a radius r_{min} are not considered as neighbors. This radius is defined as follows:

$$r_{min} = r\alpha\left(1 - \frac{1}{i}\right), \tag{3}$$

where i is the iteration number and α is a coefficient in $[0, 1]$. The higher α is, the higher the radius r_{min} is, and therefore, the higher the connection lengths are. The connection linked x_{new} and $x_{nearest}$ is nevertheless kept to preserve the optimal connection. Figure 2 shows this change, only the points in the "donut" zone (in green) are considered neighbors. In this example, the second closest neighbor is therefore not connected.

The second modification of the RRG algorithm is on the sampling method. Most of the time, the samples are randomly generated in the whole space. To avoid obtaining too long trajectories, it is proposed to sample around the straight line linking the start and final positions. Figure 3 shows the proposed sampling area. This area is defined by the distance d_s to the straight line computed as follows:

$$d_s = \frac{d}{\beta}\left(1 - \sqrt{\frac{|d_o - d/2|}{d/2}}\right), \tag{4}$$

[5] $\mu([a1, b1] \times [a2, b2]) = (b1 - a1) * (b2 - a2)$ where $b1 > a1$ and $b2 > a2$.
$\mu([a1, b1] \times [a2, b2] \times [a3, b3]) = (b1 - a1) * (b2 - a2) * (b3 - a3)$ where $b1 > a1$, $b2 > a2$ and $b3 > a3$.

[6] In 2D space, ζ_2 is the surface of a disk of radius 1 ($\zeta_2 = \pi$).

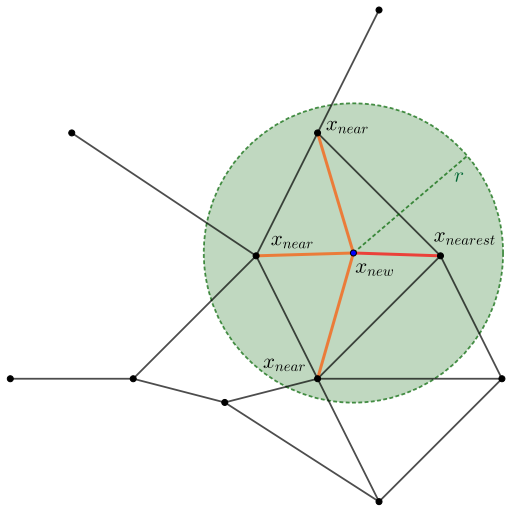


Fig. 1: RRG local optimization iteration: First, a connection linked x_{new} and its nearest neighbor $x_{nearest}$ is created (in red) and then, x_{new} is connected to the points x_{near} at a distance lower than r (in green).

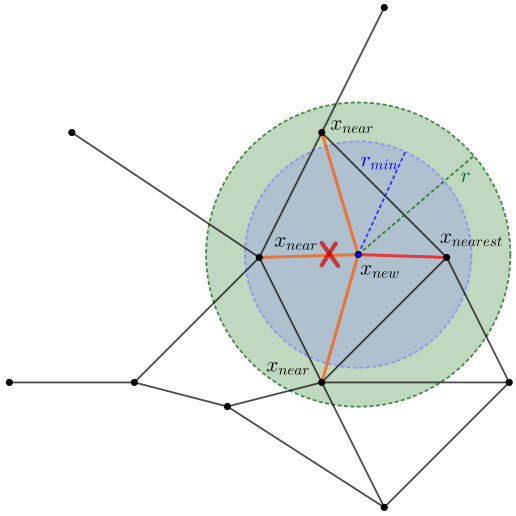


Fig. 2: RRG local optimization iteration with "donut": Connection linked x_{new} and $x_{nearest}$ is created as usual but the edge with the second closest point is not added to the graph.

where d is the distance from the origin O to the destination D, d_o is the distance on the straight line from the origin and β is a coefficient in $\mathbb{N}^*$.

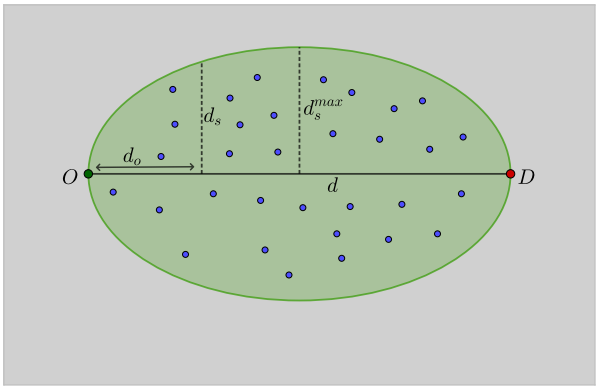


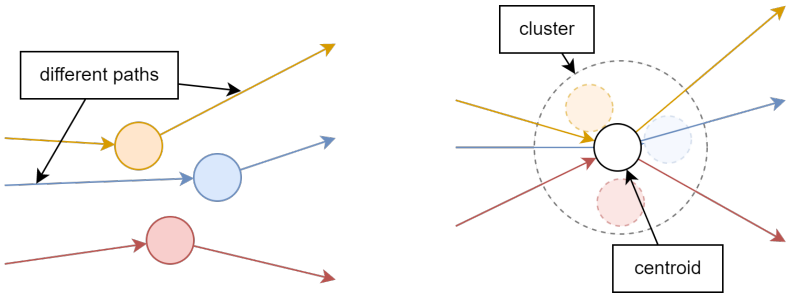
Fig. 3: The sampling area (in green) is defined around the straight line linking the origin O and the destination D by the distance d_s.

This sampling reduces the graph propagation area. The number of samples can thus be lower and thus reduce the computation time of the graph.

3.2 Clustering

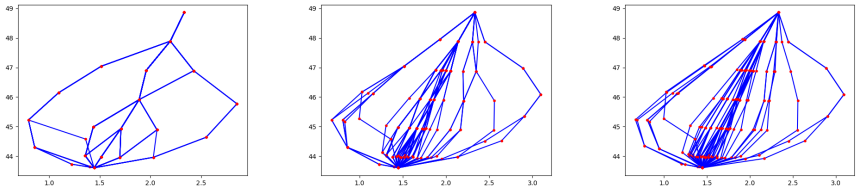
After generating all the alternative paths by the RRG algorithm, we remark that a lot of nodes were really close to each other (Figure 4a). Having small groups of nodes gathered in small size space zone is not that relevant. For instance, if one node is in a weather zone, its closest nodes will also be in this area. Therefore, closest nodes are merged into only one centroid node (Figure 4b). Performing this step is also beneficial for the post treatment on the similarity.

We choose a k-means algorithm for the clustering processing because it is an unsupervised method with only one parameter which is the number of clusters. Besides, there are no outliers in the data. Indeed, all the points belong to at least one path because they have been generated by the RRG algorithm. Figure 5 shows a run on a Paris-Toulouse trip for three sizes of clusters (20, 70, and 130). When the number of clusters is low (Figure 5a), distant points can be grouped together as it is in the top right corner and in the bottom left corner. For 70 clusters (Figure 5b), the paths are smoother than the previous clustering and there are less overlapping after replacing the points by their centroids. On Figure 5c, the paths are more precise but also more similar because the clusters are smaller. A trade-off has to be done here between similarity and precision. Because this step is followed by a similarity post treatment, we can focus on precision.



(a) Before clustering (b) After clustering

Fig. 4: Three spatially close nodes from three different paths merged into one centroid before/after clustering



(a) Clustering with 20 clusters (b) Clustering with 70 clusters (c) Clustering with 130 clusters

Fig. 5: K-means clustering of the points generated by RRG algorithm for three different number of clusters (20, 70 and 130) for the Paris-Toulouse trip

3.3 Similarity post processing

The RRG algorithm generates lots of paths between origin and destination nodes. The clustering step merged very close points into centroids. However, some of the paths generated shared too many common links. Too similar paths (red paths on Figure 6), will not allow a correct avoidance maneuvering of a weather obstacle by an aircraft because both paths will go through such a zone. These potential alternatives are finally not really alternatives. The idea behind this process is therefore to detect similar alternative paths among those generated and merge them.

The similarity metric used in this paper is Dynamic Time Warping (DTW). This method can compare two temporal sequences with potentially different speeds, and sizes. The algorithm computes the optimal matches between these two sequences. Let us consider two paths $\mathbf{x} = (x_i)_{0 \leq i \leq n}$ and $\mathbf{y} = (y_j)_{0 \leq j \leq m}$ connecting a departure airport to an arrival airport i.e. $x_0 = y_0$ and $x_n = y_m$. The DTW of $\mathbf{x}$ and $\mathbf{y}$ is defined by:

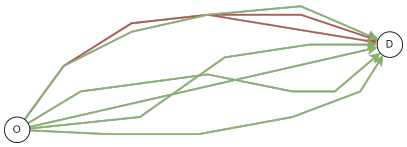


Fig. 6: Post treatment on generated paths by RRG algorithm to detect and remove similar paths (red paths) and keep diversity paths (green paths)

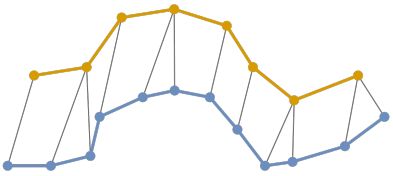


Fig. 7: DTW point matching process between two trajectories (blue and orange) with different lengths

$$DTW(\mathbf{x},\mathbf{y}) = min_{\mu\in\mathcal{M}_{\mathbf{x},\mathbf{y}}} \sum_{(i,j)\in\mu} d(x_i, y_j), \tag{5}$$

with $\mathcal{M}_{\mathbf{x},\mathbf{y}}$ being the set of possible matching between paths $\mathbf{x}$ and $\mathbf{y}$. d is a distance between two points.

The solution algorithm is based on dynamic programming. It computes the best pair matching between the points of both trajectories greedily based on the distance (Figure 7). The similarity of these trajectories is the sum of these best pair matching distances.

3.4 Proposed algorithm

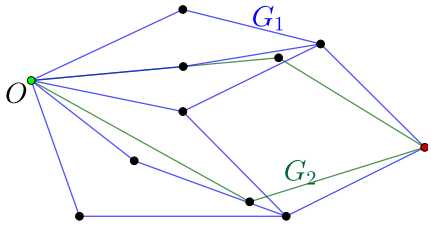
After presenting the different steps of the method, this section summarizes the complete alternative trajectories generation process (See Figure 8 and Algorithm 1). The procedure is the following:

1. n_g graphs are generated with different values of maximum neighborhood radius η (Lines 2 to 5). This process computes short and long avoidance maneuvers to avoid small or big obstacles. An example with 2 graphs is given in Figure 8a.
2. The n_g generated graphs are then clusterized to obtain a graph with more edges (Line 6). As explained in Section 3.2, a lot of nodes are very close. Close nodes are therefore merged to obtain only one graph (See Figure 8b).
3. A post-process based on DTW is done to remove similar paths (See Figure 8c and Line 7).
4. If an obstacle appears, the paths passing through the obstacle are removed from the graph to keep only those avoiding it (See Figures 8d and 8e and Lines 8 to 10).

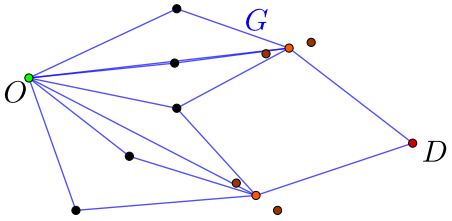
The first three steps are performed before the flight while computing the optimal flight plan. The last step can be used in different ways. It can be done when an unexpected event occurs (weather obstacle for instance). It can also be used to change the optimal flight plan by taking into account the contrail areas. This option is discussed in the next section.

Algorithm 1 Alternative paths generation algorithm

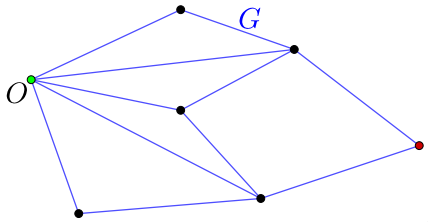
Require: $n \in \mathbb{N}^*$, $n_g \in \mathbb{N}^*$, $\mathrm{H} = \{\eta_i : \eta_i > 0,\ i \in [\![1, n_g]\!]\}$
$\mathcal{G} \leftarrow \emptyset$
for $\eta \in \mathrm{H}$ **do**
  $G = RRG(\eta, n)$
  $\mathcal{G} \leftarrow \{G\} \cup \mathcal{G}$
end for
$G \leftarrow$ `clusterize`$(\mathcal{G})$
$\mathcal{P} \leftarrow$ `DTW`(G)
if `obstacles()` **then**
  $\mathcal{P} \leftarrow$ `avoidancePaths`$(\mathcal{P})$
end if
return $\mathcal{P}$



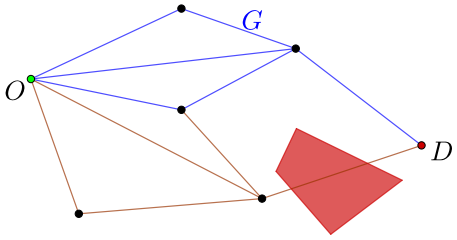
(a) Graphs generation: G_1 is generated with a small radius and G_2 with a long one.



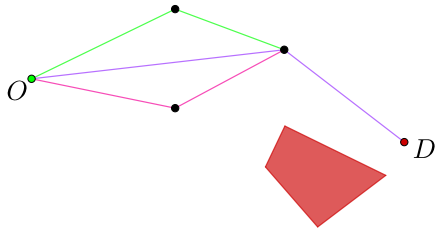
(b) Graphs clusterization: points are merged to create new points in orange.



(c) Removal of similar paths



(d) Removal of paths passing through the obstacle



(e) Selection of avoidance paths

Fig. 8: Alternative trajectories generation process

4 Results

In this paper, it is proposed to apply the alternative trajectories generation to the contrails avoidance problem. Contrail zones are not considered as "hard" obstacles that must be avoided for aircraft safety. It would be better to avoid them to reduce the non CO_2 effects of air traffic. However, such an avoidance maneuvering would release more CO_2. A compromise based on long term impact has to be found between both options. It seems that a mix between the two options would lead to the optimal solution. Contrail zone avoidance is a complex problem, it is the reason why we focus on avoiding them like they were hard obstacles. A contrail area can be vertically or horizontally avoided. The fuel consumption is lower for short lateral avoidance than for flight level change. Moreover, in operation, flight level change is rarely performed to avoid obstacles. In this study, we only consider lateral avoidance during the cruise.

4.1 Contrails data

Relative humidity and temperature data are extracted from ECMWF [8] for two hours (6 am and 7 am UTC) on January 2, 2022, at Flight Level 300 (300 hPa pressure level). They have been extracted on a 2D-grid with a resolution of 0.1 degrees on latitude and longitude, from latitude 37.5 to latitude 55.4 and from longitude -12 to longitude 16. Based on these data, areas favorable to persistent contrails are computed thanks to the following process, as usually done in the literature (see for instance [29]). First, on a given point, the *Schmidt-Appleman criterion* is applied to determine if a contrail can be formed: a contrail is formed if the relative humidity of the air in liquid water is above a threshold $RH_w \geq r_{min}$ where

$$r_{min} = \frac{G(T - T_c) + e_{sat}^{liq}(T_c)}{e_{sat}^{liq}(T)}, \tag{6}$$

$e_{sat}^{liq}(T)$ is the saturation vapor pressure over water, T_c is the estimated threshold temperature (in Celsius degrees) for contrail formation at liquid saturation. The later is computed via:

$$T_c = -46.46 + 9.43 \log(G - 0.053) + 0.72 \log^2(G - 0.053), \tag{7}$$

where $G = \frac{EI_{H_2O} C_p P}{\epsilon Q (1-\eta)}$, $EI_{H_2O} = 1.25$ is the water vapor emission index, $C_p = 1004\ J.kg^{-1}.K^{-1}$ is the heat capacity of the air, P is the ambient pressure (in Pascals), $\epsilon = 0.6222$ is the ratio of the molecular masses of water and dry air, $Q = 43 \cdot 10^6 J.kg^{-1}$ is the specific heat of combustion, and $\eta = 0.3$ is the average propulsion efficiency of a commercial aircraft. Then, if the point is in an ice-supersaturated area, it is considered in persistent-contrail favorable area. In [29], the ice super saturated areas are determined thanks to the following criterion: $RH_i > 1$, where the relative humidity over the ice, noted RH_i is computed as follows:

$$RH_i = RH_w \cdot \frac{6.0612 \cdot \exp(\frac{18.102T}{249.52+T})}{6.1162 \cdot \exp(\frac{22.577T}{273.78+T})}, \tag{8}$$

and where T is the ambient temperature in Celsius degrees.

Figure 9 presents the persistent contrails areas of January 2, 2020, at 7 am.

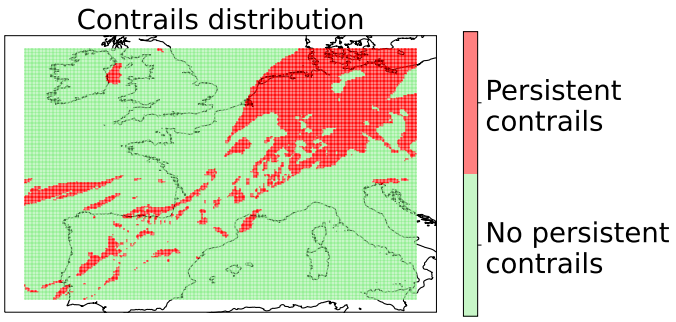


Fig. 9: Contrails data map

4.2 Alternative trajectories results

From this data, it is proposed to study two flight cases (London-Toulouse and Paris-Toulouse) to evaluate the method. In both cases, only the cruise phase is considered and the aircraft flies at FL300. Table 2 shows the parameters of the algorithm used for all tests. 15 graphs have been generated by the RRG algorithm.

Parameters	η_{min}	η_{max}	n_g	α	β	$n_{samples}$
Values	0.5	2	15	0.75	3	50,000

Table 2: Algorithm parameters.

Paris-Toulouse case As explained in Section 3, the first step is to generate several graphs. Figure 10 shows the alternative trajectories generated by the RRG with the parameters given in Table 2. In this case, 1,972 trajectories are generated. This number is too high and should be reduced. Then, the graphs are clusterized. Figure 11 shows the result of the clusterization. The number of paths remains the same but the number of nodes is significantly reduced.

Then, similar paths are removed by the similarity post-treatment. Figure 12 shows the result paths after this step. The resulting trajectories can be used

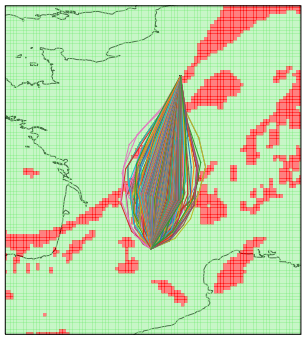


Fig. 10: Paris-Toulouse alternative trajectories generated by the RRG algorithm.

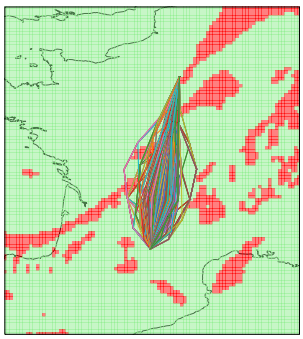


Fig. 11: Paris-Toulouse alternative trajectories after clusterization (140 clusters).

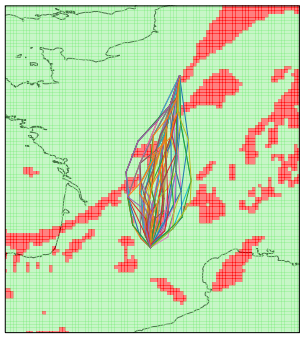


Fig. 12: Paris-Toulouse alternative trajectories after similarity post-treatment.

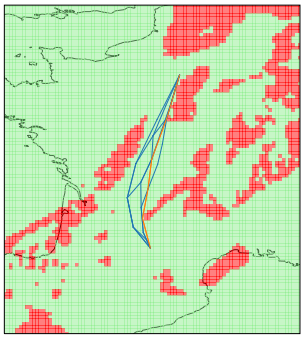


Fig. 13: Paris-Toulouse contrails avoidance trajectories at 6 am.

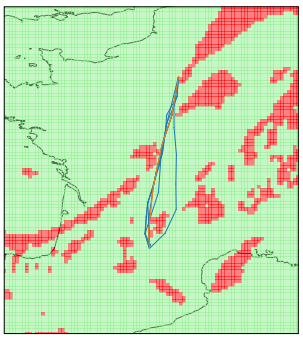


Fig. 14: Paris-Toulouse contrails avoidance trajectories at 7 am.

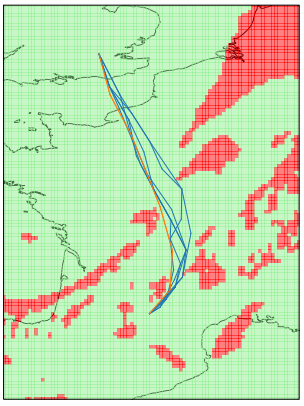


Fig. 15: London-Toulouse contrails avoidance trajectories at 7 am.

throughout the flight in case of unexpected events. It is proposed to study the use of these routes in the case of contrails avoidance.

Finally, in this case study, only contrails avoidance trajectories are kept. Figures 13 and 14 present the final paths for Toulouse-Paris at two different time scenarios. The orange path represents the shortest path from the origin to the destination. The blue paths are the alternatives. These figures show that the pilot has several options. He can first decide to follow the shortest path and then if an event occurs, he can follow a blue path just for the maneuver and then return to the optimal path. He can also choose to follow this route to the destination. Figure 14 also shows that the algorithm proposes obstacle avoidance from the right or from the left. However, it only offers one option if one maneuver is significantly longer than the other (See Figure 13). Table 3 summarizes the number of paths at each step of the proposed algorithm. Table 3 presents also the computation time of each algorithm step. The generation of the alternative trajectories takes 198.5 seconds, which is totally acceptable for a pre-flight calculation. The selection of avoidance trajectories is very fast and can be used during the flight. The simulation results show that the proposed algorithm can efficiently generate cruise alternative trajectories. Although the RRG algorithm generates a high number of paths, the KMeans and the post-processing reduce to a low but sufficient number to avoid complex obstacles.

To validate the results achieved by the methodology proposed in the paper. We compute the proportion of alternative paths connecting the Paris-Toulouse airports depending on their deviation from the shortest path (see Figure 16). This figure also includes the paths that avoid obstacles for the scenario of Figure 13. The more the alternative paths (light-blue bars) deviate from the shortest path the lesser they are. We can remark that most of the paths are close to the shortest path. Around 60% of alternative trajectories have a deviation of less than 4%. The figure shows that the obstacles can be avoided in different ways. For instance, the pilot can decide to follow the shortest avoidance path (deviation between 1% and 2%). However, he can also prefer to follow a longer trajectory to perform a safer maneuver to reduce the risk due to the uncertainties.

	RRG	KMeans	Post	Avoidance	
				6am	7am
Number of paths	1972	1972	64	8	7
Computation Time (s)	191	1	6.5	0.019	

Table 3: Number of paths and computation time in seconds of each algorithm step.

London-Toulouse case To complete this study, the London-Toulouse flight is tested. Figure 15 shows the alternative paths generated by the proposed algorithm for the London-Toulouse flight at 7 am. According to the results obtained,

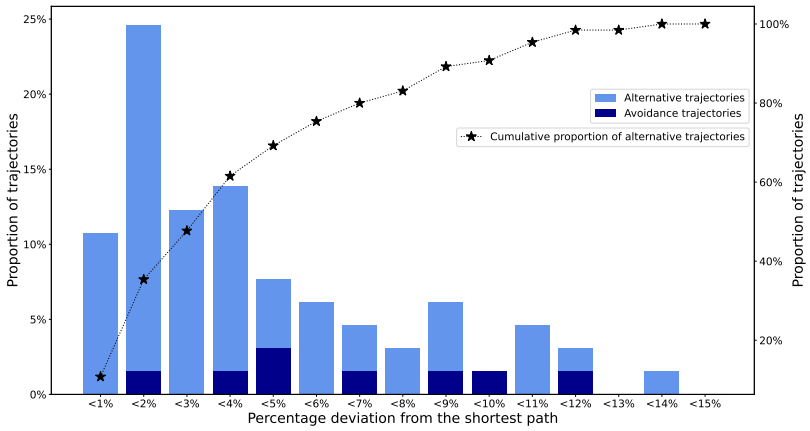


Fig. 16: Alternative and avoidance trajectories proportion depending on the deviation from the shortest path linking Paris and Toulouse. The avoidance trajectories are for the scenario of Figure 13.

the method is promising. Indeed, for several case studies, the algorithm is able to generate multiple alternative trajectories. Moreover, these trajectories can be used to avoid obstacles and offer several options to pilots.

5 Conclusion

This paper addresses the automatic generation of alternative cruise trajectories to handle any type of unexpected event. In the first part of this paper, some methods to generate alternative paths are presented. Then, a solution algorithm is proposed, based on an adapted version of the RRG algorithm, K-means clusterization, and DTW similarity metric. Simulation results based on two different flights illustrate the proposed approach. According to the numerical results, this approach generates alternative trajectories with diversity and considers some aeronautical operational constraints. An analysis of the distribution of alternative paths shows that most of the trajectories generated by our method are close to the shortest trajectory. Moreover, the method proposes avoidance trajectories with different lengths. The proposed method can be used in different ways. For instance, it can be relevant in the case of the presence of persistent condensation trails. They can also be used as an alternative to the optimal flight plan. The proposed method seems promising, but for long-haul flights, a dynamic version with updated alternative trajectories would be necessary. Moreover, vertical alternative trajectories could be added to the framework. This study opens the way to the development of a complete flight system composed of an alternative cruise trajectory system and an emergency trajectory generation tool like the

one developed in the SafeNcy project [6, 25, 26, 30]. This tool would propose alternative cruise trajectories to avoid unexpected events during a flight and trajectories in case of an emergency. Each point of the generated graph would have a set of alternatives and one or more emergency trajectories. This system could enhance the safety of the flights. The pilot could thus react to any type of event, be it a simple weather event or an emergency. This kind of tool could reduce the workload of pilots and ATCs.

Bibliography

[1] V. Akgün, E. Erkut, and R. Batta. On finding dissimilar paths. *European Journal of Operational Research*, 121(2):232–246, Mar. 2000.

[2] R. Bellman. On a routing problem. *Quarterly of applied mathematics*, 16 (1):87–90, 1958.

[3] L. Bonin, D. Delahaye, A. Guitart, E. Feron, and X. Prats. Optimal Path Planning for Soaring Flight. In *Conference on Guidance Navigation and control (CEAS EuroGNC 2022)*, Berlin, Germany, May 2022. CEAS and AIAA. URL `https://hal-enac.archives-ouvertes.fr/hal-03619377`.

[4] A. Chakravarty. Four-dimensional fuel-optimal guidance in the presence of winds. *Journal of Guidance, Control, and Dynamics*, 8(1):16–22, 1985.

[5] T. Chondrogiannis, P. Bouros, J. Gamper, and U. Leser. Alternative routing: K-shortest paths with limited overlap. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, SIGSPATIAL '15, New York, NY, USA, 2015. Association for Computing Machinery. URL `https://doi.org/10.1145/2820783.2820858`.

[6] Cordis and CleanSky. Safency - the safe emergency trajectory generator, 2020. URL `https://cordis.europa.eu/project/id/864771/fr`.

[7] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numer. Math.*, 1(1):269–271, Dec. 1959. URL `https://doi.org/10.1007/BF01386390`.

[8] ECMWF. Ecmwf datasets, 2020. URL `https://www.ecmwf.int/en/forecasts/access-forecasts/access-archive-datasets`.

[9] A. Fallast and B. Messnarz. Automated trajectory generation and airport selection for an emergency landing procedure of a CS23 aircraft. *CEAS Aeronautical Journal*, 8(3):481–492, Sept. 2017.

[10] K. Gierens, S. Matthes, and S. Rohs. How well can persistent contrails be predicted? *Aerospace*, 7(12), 2020.

[11] A. Guitart, D. Delahaye, and E. Feron. An accelerated dual fast marching tree applied to emergency geometric trajectory generation. *Aerospace*, 9(4), 2022.

[12] A. Guitart, D. Delahaye, F. M. Camino, and E. Feron. Collaborative generation of local conflict free trajectories with weather hazards avoidance. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–12, 2023. https://doi.org/10.1109/TITS.2023.3289191.

[13] X.-B. Hu, C. Zhang, G.-P. Zhang, M.-K. Zhang, H. Li, M. Leeson, and J.-Q. Liao. Finding the k shortest paths by ripple-spreading algorithms. *Engineering Applications of Artificial Intelligence*, 87:103229, 01 2020.

[14] E. A. Irvine and K. P. Shine. Ice supersaturation and the potential for contrail formation in a changing climate. *Earth System Dynamics*, 6(2): 555–568, 2015.

[15] L. Janson, E. Schmerling, A. Clark, and M. Pavone. Fast Marching Tree: a Fast Marching Sampling-Based Method for Optimal Motion Planning in Many Dimensions. Feb. 2015. URL http://arxiv.org/abs/1306.3532.
[16] M. R. Jardin and A. E. Bryson. Neighboring Optimal Aircraft Guidance in Winds. *Journal of Guidance, Control, and Dynamics*, 24(4):710–715, 2001.
[17] S. Karaman and E. Frazzoli. Incremental sampling-based algorithms for optimal motion planning, 2010. URL https://arxiv.org/abs/1005.0416.
[18] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.
[19] S. M. LaValle. Rapidly-exploring random trees : a new tool for path planning. *The annual research report*, 1998.
[20] D. Lee, D. Fahey, A. Skowron, M. Allen, U. Burkhardt, Q. Chen, S. Doherty, S. Freeman, P. Forster, J. Fuglestvedt, A. Gettelman, R. De León, L. Lim, M. Lund, R. Millar, B. Owen, J. Penner, G. Pitari, M. Prather, R. Sausen, and L. Wilcox. The contribution of global aviation to anthropogenic climate forcing for 2000 to 2018. *Atmospheric Environment*, 244:117834, 2021.
[21] K. Legrand, S. Puechmorel, D. Delahaye, and Y. Zhu. Robust aircraft optimal trajectory in the presence of wind. *IEEE Aerospace and Electronic Systems Magazine*, 33(11):30–38, 2018.
[22] L. Ligny, A. Guitart, D. Delahaye, and B. Sridhar. Aircraft Emergency Trajectory Design: A Fast Marching Method on a Triangular Mesh. In *Fourteenth USA/Europe Air Traffic Management Research and Development Seminar*, New-Orleans, United States, Sept. 2021.
[23] H. Liu, C. Jin, B. Yang, and A. Zhou. Finding top-k shortest paths with diversity. *IEEE Transactions on Knowledge and Data Engineering*, 30(3): 488–502, 2017.
[24] K. Palopo, R. D. Windhorst, S. Suharwardy, and H.-T. Lee. Wind-Optimal Routing in the National Airspace System. *Journal of Aircraft*, 47(5):1584–1592, 2010.
[25] R. Sáez, H. Khaledian, and X. Prats. Generation of emergency trajectories based on aircraft trajectory prediction. In *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*, pages 1–10. IEEE, 2021.
[26] R. Sáez, H. Khaledian, X. Prats, A. Guitart, D. Delahaye, and E. Feron. A Fast and Flexible Emergency Trajectory Generator Enhancing Emergency Geometric Planning with Aircraft Dynamics. In *Fourteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2021)*, New Orleans (virtual), United States, Sept. 2021.
[27] J. A. Sethian. A fast marching level set method for monotonically advancing fronts. *Proceedings of the National Academy of Sciences*, 93(4):1591–1595, 1996.
[28] J. A. Sethian and A. Vladimirsky. Ordered upwind methods for static Hamilton–Jacobi equations. *Proceedings of the National Academy of Sciences*, 98(20):11069–11074, 2001.
[29] M. Soler, B. Zou, and M. Hansen. Flight trajectory design in the presence of contrails: Application of a multiphase mixed-integer optimal control

approach. *Transportation Research Part C: Emerging Technologies*, 48:172–194, Nov. 2014.

[30] R. Sáez, I. Octavian Rad, X. Prats, B. Viry, P. Gonzalez, A. Guitart, D. Delahaye, O. Larrieu, and K. Rebaï. An automated emergency airport and off-airport landing site selector. In *2022 IEEE/AIAA 41st Digital Avionics Systems Conference (DASC)*, pages 1–10, 2022. https://doi.org/10.1109/DASC55683.2022.9925757.

[31] L. Talarico, K. Sörensen, and J. Springael. The k-dissimilar vehicle routing problem. *European Journal of Operational Research*, 244(1):129–140, 2015.

[32] J. Y. Yen. Finding the k shortest loopless paths in a network. *Management Science*, 17(11):712–716, 1971.