A new method for generating optimal conflict free 4D trajectory

Nour Elhouda Dougui, Daniel Delahaye, Stéphane Puechmorel, Marcel Mongeau

A new method for generating optimal conflict free 4D trajectory

Nour Dougui
Applied Mathematics Laboratory
ENAC
7, Avenue Edouard Belin
31055 Toulouse, France
Email: nour@recerche.enac.fr

Daniel Delahaye
and Stephane Puechmorel
Applied Mathematics Laboratory
ENAC
7, Avenue Edouard Belin
31055 Toulouse, France
Email: delahaye@recerche.enac.fr
Email: puechmor@recerche.enac.fr

Marcel Mongeau
Université de Toulouse UPS, INSA, UT1, UTM
Institut de Mathématiques de Toulouse
F-31062 Toulouse cedex 9, France
CNRS
Institut de Mathématiques de Toulouse UMR 5219
F-31062 Toulouse cedex 9, France
Email: mongeau@math.univ-toulouse.fr

***Abstract*—The need for increasing air traffic capacity motivates 4D trajectory planning concept. In order to generate conflict-free 4D trajectories, we introduce a new concept based on light propagation modeling algorithm. This algorithm is a wavefront propagation method that yields a natural solution for the path planning problem specifically in the case of air traffic congestion.**

I. Introduction

The analysis of air traffic growth expects a doubling of the number of flights over the next 20 years. The Air Traffic Management (ATM) will therefore have to absorb this additional burden and increase the airspace capacity, while ensuring at least equivalent standards of safety and interoperability. The European project SESAR was initiated to propose solutions to this problem. It relies on a new concept of air traffic control, known as 4D (3D plus time) trajectory planning, which consists in exploiting the possibilities of the Flight Management System (FMS) to ensure that a given aircraft is at a given position at a given time. For each flight, a reference trajectory, called Reference Business Trajectory (RBT), is requested by the operating airline. During the flight, conflict situations may nevertheless occur, in which two or several aircraft can dangerously approach each other. In this case, it is necessary to modify one or more trajectories to ensure that minimum separation standards (currently 5 Nm horizontally and 1000 ft vertically) are still satisfied. Moreover, it is desirable that proposed new trajectories deviate as little as possible from RBT. In this context, we propose a new algorithm which seeks to ensure sufficient separation between aircraft while producing flyable trajectories.

A. Previous related works

During recent years, several methods have been proposed to find an optimal solution that could solve conflicts in air traffic. The aim of these methods is to find for each aircraft, an optimal 4D trajectory that avoids conflicts with other aircraft, reaches the destination point and optimizes a cost function which depends on the travel duration and on the cost index (a coefficient that takes into account fuel consumption). There are mainly two classes of methods to address this problem: genetic algorithms [1] and navigation-function based approach [2]. Each one provides only a partial solution to the problem.

The first one, genetic algorithms, consists in generating a new population of aircraft trajectories from a base population using three basic operators: selection, mutation and crossover in order to improve the cost function. This process is iterated until the cost function is no longer improved. The state space is a set of finite maneuvers, which are straight lines, turning points (changing an aircraft heading and then bringing it back on its initial trajectory) and offsets (inducing a lateral shift from the initial trajectory). Those maneuvers are the ones used by air traffic controller. Genetic algorithms generate trajectories with feasible operational maneuvers and with velocities within bounded ranges. They can reach asymptotically optimal solution, but for a given computing time, a feasible (conflict-free) solution is not guarantee.

The second method, based on navigation functions, consists in using an electrostatic modeling of the problem: an electron (which has a negative charge) is subject to an electric field, and is attracted by a positive charge which represents the goal and is pushed away by negative charges which represent obstacles. Thus, the electron is going to move towards the goal and steered by the resultant electric fields. The aircraft (a virtual electron) is represented by a point in 3D space. If a mathematical function of potential fields can be built to model adequately the destination charge and the distribution of the obstacle charges, then the virtual forces applied on such virtual electron, initially positioned at the departure point, can be computed. This produces a trajectory which connects the departure point with the destination while avoiding obstacles (the other aircraft).

Navigation functions have already demonstrated their effectiveness in motion planning with guaranteed collision avoidance and convergence towards the goal configuration (reach the destination point with the right orientation). However, they do not take into account the constraints imposed by ATM, such as bounded speed, smooth trajectory and time constraints. Besides, they may tolerate large deviations from RBT.

The objective of our approach, based on an optical analogy, is to find for each aircraft a feasible (relevant to ATM constraints) optimal 4D trajectory, avoiding conflicts and which minimizes a criterion based on a local metric.

B. Paper overview

In the next section, we present our method. Numerical results are presented in section III. Then, a conclusion appears in section IV.

II. Light Modelling Algorithm

In order to build our algorithm, a light propagation analogy is used. In the physical framework, light propagates in space under *Decarte laws* (see Figure 1).

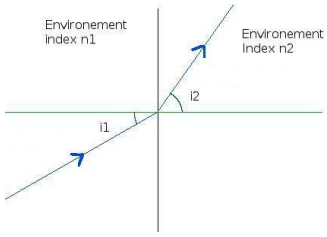

Fig. 1. Light ray deviation under Decarte laws in a region cut into two index areas (n1,n2)

These laws are summarized by the following:

For a light ray that goes from a region with index n_1 into an other one with index n_2 with an angle i_1, we have

$$\begin{cases} n_1 \quad sin(i_1) = n_2 \quad sin(i_2) \\ v \quad = \frac{c}{n2} \end{cases}$$

where, i_2 is the angle of the light ray in region with index n_2, v its associated velocity and c the velocity of the light in vacuum.

Our algorithm uses such laws in order to build aircraft trajectories using congestion or other aircraft neighborhood as high index areas.

The light modelling algorithm is adjusted from the aircraft point of view. It is assumed that aircraft knows the surrounding aircraft trajectories (trajectories of other aircraft is a given input of our algorithm).

Assume that objective function is an application associating a positive real value to a curve of class C^1 of $\mathbb{R}^3$. Such value is computed by integrating a local metric along the curve. We can thus represent length, travel time or the cost associated with a trajectory by a suitable choice of local metric. Determining an optimal trajectory will therefore reduce to search a geodesic which is the shortest path between two points on the space regarding the local metric.

In this algorithm, we use the well-known fact that a light ray trajectory is a geodesic when considering the environment refractive index as a local metric. To represent congestion areas and conflicts in air traffic management, we consider the refractive index as a measure of congestion or traffic complexity. We select a barrier index value in the prohibited areas and in the protection volumes surrounding each aircraft. The optimal trajectory will be computed using a technique of ray tracing. The light will be slowed down in congested areas, but despite this, it can pass through. However, it will be completely blocked by aircraft protection volumes, which ensures conflict free-situations. We launch several light rays in various directions from the departure point of the aircraft. The path of the first ray that reaches the arrival point corresponds to an approximation of a geodesic.

We compute the environment index associated to a given congested area using a model based on Lyapunov exponents [3].

To generate a trajectory, we use a wavefront propagation algorithm in 3D with a space discretization (the wave propagation is done with a space step ds) from the departure point. We do not propagate the wavefront randomly in all space directions but into directions with highest probability of success. To ensure this, we guide the wavefront by an initial solution obtained by the navigation function method [2]. Consequently, we can guarantee at least one feasible solution. To avoid a combinatorial explosion, the propagation will be coupled with a branch-and-bound algorithm that interrupts unnecessary shooting rays. In our case, the trajectory obtained by the navigation function method is sampled with half-spheres of radius ds, oriented towards the destination. These half-spheres are, in turn, sampled with an angle step $d\theta$ in the horizontal plane and an angle step $d\varphi$ in the vertical plane to build an initial search tree for the branch-and-bound algorithm as shown in Figure 2.

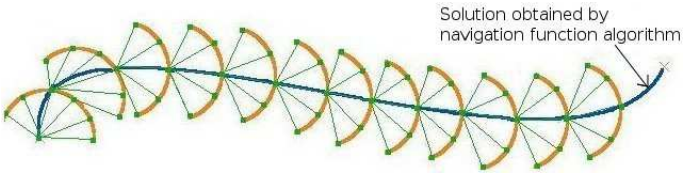

Fig. 2. Initial search tree for the branch-and-bound algorithm

A. Branch-and-bound Algorithm

We implement the propagation within a *branch-and-bound algorithm* [4], a classical framework for solving discrete optimization problems. At the beginning, we consider the set of all possible solutions, represented by the root. Procedures to obtain lower and upper bounds for the optimal value of our criterion are applied to the root. If these two bounds are equals, then the optimal solution is found, and the algorithm stops. Otherwise, the solution set is partitioned into two or more sub-problems, which become children of the root. The method is then applied recursively on these sub-problems, generating a tree.

The idea behind the building of sub-problems which are relaxations of the original problem, branching process, is to solve them in a reasonable time. If an optimal solution is found for a sub-problem, it is feasible but not necessarily optimal for the original problem. On the other hand, as a feasible solution,

it can be used to eliminate partial solutions. Indeed, if the lower bound of a node exceeds the value of an already known solution, then we can say that global optimum solution cannot belong to in the subset of solution represented by this node. It is therefore eliminated. The search goes on until all the nodes are explored or eliminated.

B. Branch and Bound applied to the Light Modeling Algorithm

The initial upper bound used in the Branch and Bound algorithm is the travel time computed by the navigation function method.

In order to build a lower bound for a given search tree node, we first compute a bound, "*TimeToDest*", for the remaining time to reach the destination. This bound is a weighted sum of two terms. The first one, "*integTime*", is the summation of the refractive index along the direct route to the destination. The second one, "*maxSpeedTime*", is the time needed to reach destination in direct route with the maximum speed.

$$TimeToDest := \alpha * integTime + \beta * maxSpeedTime. \quad (1)$$

with weighting parameters α, β such that $\alpha + \beta = 1$.

The lower bound is then the summation of *TimeToDest* and the time needed to reach the node from the origin (*TimeToNode*). More specifically, the lower bound is given by (see Figure 3):

$$lowerBound := TimeToNode + TimeToDest.$$

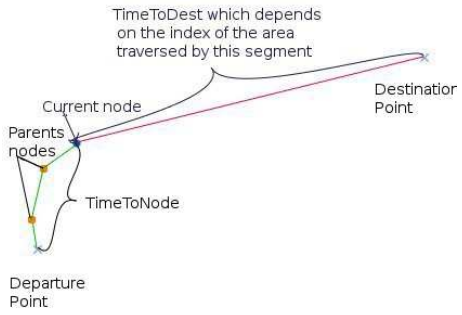

Fig. 3. The lower bound computing.

Branching, in our context, involves launching rays as straight lines in a spatial half sphere of radius ds oriented towards the arrival point.

Browsing the search tree can be done in different ways. We choose a strategy whose priority is to find quickly a feasible solution (depth-first search or DFS). Here a live node (a node for which children have not yet been generated) with deepest level in the search tree is chosen for exploration. The memory requirement in terms of number of subproblems (stored at any given time) is bounded above by the number of levels in the search tree multiplied by the maximum number of children of any node, which is in our context a manageable number. The drawback of such approach is that nodes which are far from being optimal, may yield large amount of unnecessary bounding computations.

In order to avoid such drawback in our case, DFS is combined with a selection strategy. This consists in selecting the node that has the best lower bound among the nodes at the same level in the search tree (a combination of DFS as the overall principle and best first search, BeFS as a secondary selection criterion).

At each time step dt of the algorithm, the environment index is updated because aircraft change themselves the congestion while moving.

The main steps of the algorithm are as follows:

1. Compute the navigation function algorithm solution: trajectory T. Set UpperBound := time travel of T.
2. Descritize T to build a tree, with half spheres having radius ds, an angle step $d\theta$ horizontally and an angle step $d\varphi$ vertically.
3. Set TrajSolution := null. While the destination is not reached do:
   a. Update the refractive index.
   b. While there is still unexplored nodes in the tree do:
      - Choose a node N.
      - Relaunch rays from node N:
        For any light ray, if the light beam goes from a region with index n_1 into a region with index n_2 with an angle i_1, let it continue with a new angle i_2 such that n_1 $\sin(i_1)$ = n_2 $\sin(i_2)$ and with a velocity of $v = \frac{c}{n_2}$.
   c. Set TrajSolution := Trajectory obtained at b. interrupted by the step time dt.

In the following section, we will see the numerical results of a simplified version of the algorithm (2D), first with a static refractive index and then with a dynamic refractive index.

III. Numerical Results

In all our simulations, we work on a 3.2 GHz machine running under Debian Linux operating system, 1024 KB of RAM. The software has been developed in JAVA.

We use a coordinate system that is scaled with separation standards. Thus, we use an (x, y) grid with a standard horizontal separation (5 Nm) unit. The index map used is a square of $(15 * 15)$ standard horizontal separation.

In step 2. of the algorithm, the radius ds of the semicircles is set to a half standard separation distance. And the sampling angle $d\theta$ is set to $\frac{\Pi}{10}$. The coefficients in the formula (1) we chose, are: $\beta := 0.1$ and $\alpha := 0.9$.

A. Results in 2D

The algorithm was first tested with a static refractive index function (it does not depend on time) in 2D space in order to highlight the fact that it does find geodesics in simple cases.

Several refractive index functions were tested. Some examples are presented in Figures 4, 5, 6 and 7. The index

function used is a continuous function, where high values are represented in red and low values in blue. Thus, the congested areas are represented in red and areas that involve little traffic are shown in blue. The resulting solution trajectory is found in less than 5 s of CPU.

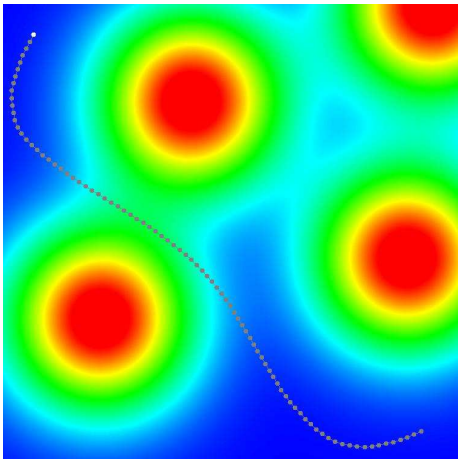

Fig. 4. Resulting trajectory with departure point on the bottom right corner and arrival point on the top left corner. The function index is given by: $e^{-((x-a_1)^2+(y-b_1)^2)/k} + e^{-((x-a_2)^2+(y-b_2)^2)/k} + e^{-((x-a_3)^2+(y-b_3)^2)/k} + e^{-((x-a_4)^2+(y-b_4)^2)/k}$

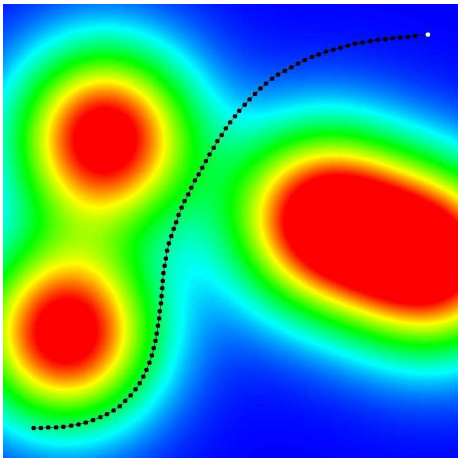

Fig. 5. Resulting trajectory with departure point on the bottom left corner and arrival point on the top right corner. The function index is given by: $e^{-((x-a_1)^2+(y-b_1)^2)/k} + e^{-((x-a_2)^2+(y-b_2)^2)/k} + e^{-((x-a_3)^2+(y-b_3)^2)/k} + e^{-((x-a_4)^2+(y-b_4)^2)/k}$

In Figures 4, 5 and 6, trajectories avoid high index area and passe through "valleys" which is the expected behavior. Thus, the aircraft avoids congested areas.

In Figure 7, the trajectory goes through a relatively congested area instead of bypassing it completely through the blue area above. This behavior can be explained by the fact that direct path, although it slows down the aircraft is more advantageous than a long detour with a higher aircraft velocity.

In all cases, the resulting trajectory is a geodesic approximation.

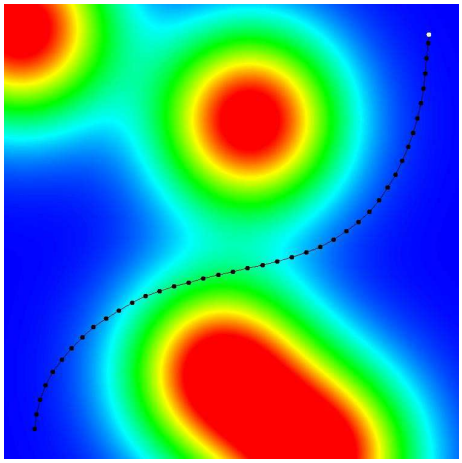

Fig. 6. Idem Figure 5 with new parameters $(a_i, b_i, ii \in \{1..4\})$ for the index function.

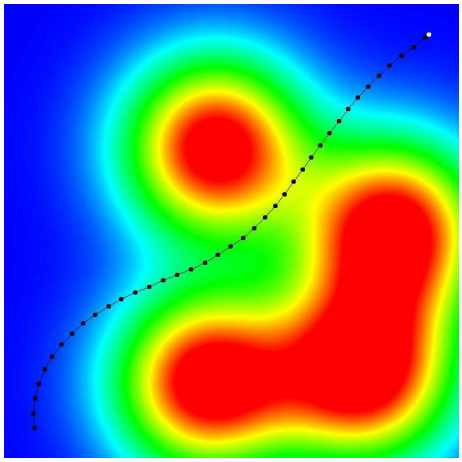

Fig. 7. Idem Figure 5 with new parameters $(a_i, b_i, ii \in \{1..4\})$ for the index function.

B. Results in 2D + time

Here, we deal with a situation of conflict resolution involving several aircraft. The algorithm controls one aircraft trajectory and we assume that the other aircraft maintain fixed straight line trajectories. The refractive index takes into account two factors. The first one is related to avoidance of the protection zones of other aircraft. The second one drive away the aircraft from congested areas where it could be in conflict.

The index function n is given by the following formula:

Consider P aircraft $(a_i)_{i\in\{1..P\}}$ moving in the space with the associated position vectors $(\overrightarrow{X_i})_{i\in\{1..P\}}$ and velocity vectors $(\overrightarrow{V_i})_{i\in\{1..P\}}$. For any space point $\overrightarrow{Y}$,

$$n(\overrightarrow{Y}) := n_1(\overrightarrow{Y}) + C(\overrightarrow{Y}).$$

The first function n_1 is used to avoid conflict. In order to ensure that the aircraft controlled by the algorithm avoids the other aircraft, we represent them by disks whose radius is

the standard distance separation. We set the function n_1 to a very high value N inside these disks and we make it decrease rapidly outside the disk as follows:
For any space point $\overrightarrow{Y}$, let

$$||\overrightarrow{X_i} - \overrightarrow{Y}|| := \alpha.$$

$$\begin{cases} \alpha \leq R & \Rightarrow \quad n_1(\overrightarrow{Y}) := N & N \gg 1. \\ \alpha \geq R & \Rightarrow \quad n_1(\overrightarrow{Y}) := 1 + \frac{N-1}{1+(\alpha-R)^q}. \end{cases}$$

with R the standard distance separation and q is a parameter that determines the speed with which the index decreases outside the separation zone.

The second function C called *convergence metric* [5] models the fact that aircraft converge to a point in the space. It is a metric used to mesure congestion. The *convergence value* at a space point $\overrightarrow{Y}$ is given by the formula:

$$C(\overrightarrow{Y}) := \sum_i c(i) * e^{-\beta*(||\overrightarrow{X_i} - \overrightarrow{Y}||)},$$

where

$$c(i) := \sum_{\substack{j \neq i, \\ r_{ij} < 0}} r_{ij} * e^{-\alpha*(||\overrightarrow{X_i} - \overrightarrow{X_j}||)}$$

where

$$r_{ij} := \frac{(\overrightarrow{X_i} - \overrightarrow{X_j})}{||\overrightarrow{X_i} - \overrightarrow{X_j}||} * (\overrightarrow{V_i} - \overrightarrow{V_j})$$

and where α and β are weighting parameters.

Some examples of conflict resolutions with 2 to 4 aircraft are shown in Figures 8, 9, 10, 11 and 12.

The track of the aircraft controlled by the algorithm appears in blue. This aircraft has its starting point at the bottom right corner of the figure and its arrival point at the top left corner. Its initial trajectory (a straight line) is shown in red. The other aircraft trajectories appear in black. The starting point of each trajectory is represented by a small dot.

In all these cases, the solution is found in less than 30 s of CPU.

In Figure 8, only two aircraft are in conflict. Their trajectories cross at the center of the figure. The controlled aircraft avoids conflict and follows a path that consists in two segments and an arc.

In Figures 9, 10 and 11, three aircraft are involved. The controlled aircraft is initially in conflict with one or the two other aircraft.

And each time, the trajectory generated by the algorithm avoids conflicts while remaining close enough to the original trajectory.

Moreover, these trajectories are composed of a set of segments and arcs.

In Figure 12, four planes are considered. As previously, the resulting trajectory avoids conflicts by using 3 segments and an arc.

Figure 13 represents a situation with four aircraft in conflict. The resulting trajectory successfully avoids conflicts, but the turns used are too steep. This occurs because the metric used for the index (convergence indicator) does not have a big enough predictability horizon. This creates paths that are not quite regular. We will therefore seek for a better congestion metric that avoids the occurence of such irregularities.

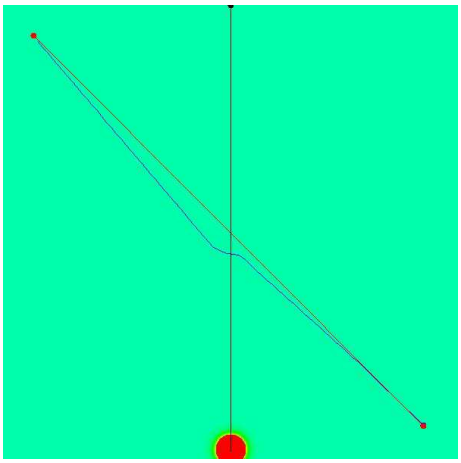

Fig. 8. Conflict resolution with 2 aircraft. Controlled aircraft initial trajectory is in red with a departure point on the bottom right corner and an arrival point on the top left corner. Resulting trajectory is in blue. The other aircraft trajectory starts on the top in the middle and goes to the bottom. Its separation zone is represented by a red disk.

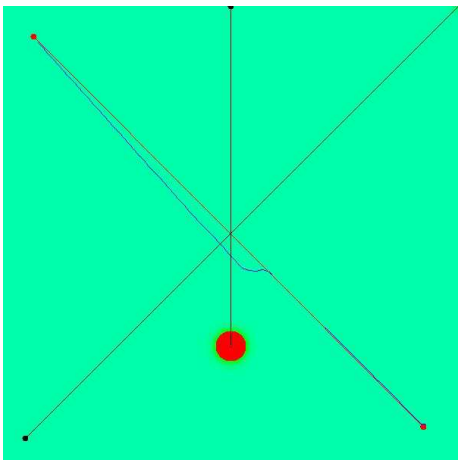

Fig. 9. Conflict resolution with 3 aircraft. Controlled aircraft initial trajectory is in red with a departure point on the bottom right corner and an arrival point on the top left corner. Resulting trajectory is in blue. The first conflicting aircraft trajectory starts on the top in the middle and goes to the bottom. The second conflicting aircraft trajectory starts on the left bottom corner and goes to the right top corner.

IV. Conclusion

The trajectory produced with our light-model algorithm avoids conflicts. It is a geodesic approximation that guarantee a speed lower bound, which is critical for an aircraft. Moreover, this trajectory is a sequence of segments and arcs, a trajectory that the FMS can monitor. Indeed, in most cases, it is a smooth curve which is flyable and which apparently does not require a lot of RTA points (Required Time on Arrival. This means

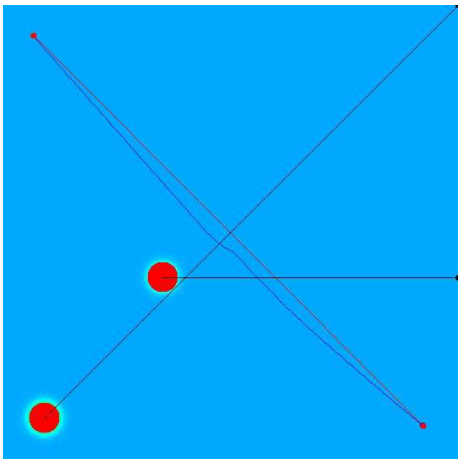

Fig. 10. Conflict resolution with 3 aircraft. Controlled aircraft initial trajectory is in red with a departure point on the bottom right corner and an arrival point on the top left corner. Resulting trajectory is in blue. The first conflicting aircraft trajectory starts on the right and goes to the left. The second conflicting aircraft trajectory starts on the right top corner and goes to the left bottom corner.

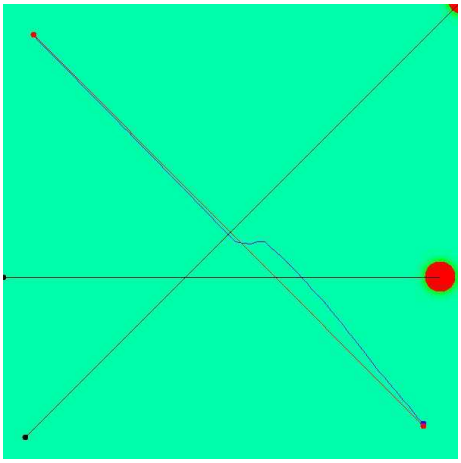

Fig. 11. Conflict resolution with 3 aircraft. Controlled aircraft initial trajectory is in red with a departure point on the bottom right corner and an arrival point on the top left corner. Resulting trajectory is in blue. The first conflicting aircraft trajectory starts on the left and goes to the left. The second conflicting aircraft trajectory starts on the left bottom corner and goes to the right top corner.

that the plane must be in a given position at a given moment). Indeed, RTA points correspond to curvatures changes and in the tested examples, not a lot of curvatures changes appear.

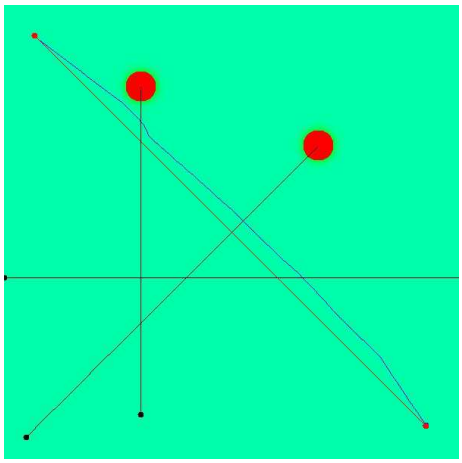

Fig. 12. Conflict resolution with 4 aircraft. Controlled aircraft initial trajectory is in red with a departure point on the bottom right corner and an arrival point on the top left corner. Resulting trajectory is in blue. The first conflicting aircraft trajectory starts on the left and goes to the left. The second conflicting aircraft trajectory starts on the left bottom corner and goes to the right top corner. The third conflicting aircraft trajectory starts from the bottom and goes to the top.

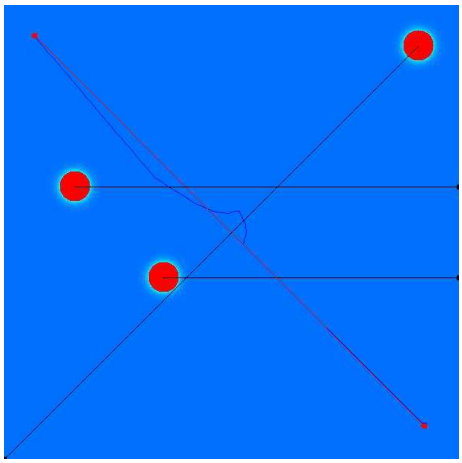

Fig. 13. Conflict resolution with 4 aircraft. Controlled aircraft initial trajectory is in red with a departure point on the bottom right corner and an arrival point on the top left corner. Resulting trajectory is in blue. The first conflicting aircraft trajectory starts on the left bottom corner and goes to the right top corner. The two other conflicting aircraft trajectory starts from the right and goes to the left.

In the next step, a better convergence indicator will be used. The curvature of the resulting trajectories will be analyzed in order to check it satisfy aircraft constraints. This approach allows us to know how many RTA points are required to monitor conflict-free trajectories. Current work involves testing a new method for generating geodesic curves on a triangular mesh following ideas from computer graphics[6]. Our aim is to improve our results both in term of geodesic approximation and in term of computational time.

REFERENCES

[1] N. Durand. *Optimisation de trajectoires pour la rsolution de conflits en route.* PhD thesis, ENSEEIHT, Institut National Polytechnique de Toulouse, France, 1996.

[2] G. Roussos, G. Chaloulos, K. Kyriakopoulos and J. Lygeros. *Control of multiple non-holonomic air vehicles under wind uncertainty using model predictive control and decentralized navigation function*, IEEE Conference on Decision and Control, December, 2008.

[3] S. Puechmorel and D. Delahaye. *Dynamical systems complexity with a view towards air traffic management applications*, IEEE Conference on Decision and Control, 2009.

[4] E. Balas and P. Toth. *Branch and Bound Methods, In The Traveling Salesman Problem*, John Wiley & Sons, 361-401, 1985.

[5] D. Delahaye, S. Puechmorel. *Air traffic complexity: towards intrinsic matrics*, 3rd USA/Europe Air Traffic Management R& D Seminar Napoli, 2000.

[6] M. Novotni, R. Klein. *Computing geodisic distances on triangular meshes*, The 10th International Conference in Central Europe on Computer Graphics, 2002.