A Light-Propagation Model for Aircraft Trajectory Planning

Nourelhouda Dougui Daniel Delahaye
Stéphane Puechmorel Marcel Mongeau.

Received: date / Accepted: date

Abstract

Predicted air traffic growth is expected to double flight number over the next 20 years. If current means of air traffic control are maintained, airspace capacity will reach its limits. The need for increasing airspace capacity motivates improved planning aircraft trajectories in 4D (space+time). In order to generate sets of conflict-free 4D trajectories, we introduce a new nature-inspired algorithm: the light propagation algorithm (LPA). This algorithm is a wavefront propagation method that yields geodesic solutions (minimum-in-time solutions) for the path planning problem, particularly in the case of air-traffic congestion. In simulations, LPA yields encouraging results on real-world traffic over France while satisfying the specific constraints in air-traffic management.

0.1 Introduction

When flying between two airports, aircraft must follow a specified set of routes and *beacons* (crossing points of several airways) on an airway network in order to structure the traffic for the ease of the air traffic controllers. As a consequence, beacons promote conflicts between aircraft when trajectories converge on common points of the network structure, thereby inducing a risk of collision. Air traffic controllers monitor traffic and ensure that aircraft follow their planned trajectories. When two or more aircraft are converging to the same point, controllers are required to issue heading, speed or altitude manoeuvers to some aircraft in order to ensure a minimum distance (i.e. *separation distance*) between all aircraft. For en-Route cruising aircraft, air traffic controllers usually use heading changes for conflict resolution in order to minimize the impact on flight efficiency (vertical manoeuvers are mainly used for terminal area); moreover heading changes have a more effective impact on the traffic situation from the air traffic controller point of view (compared to speed regulation). Speed control is effective for conflict resolution only in the descent phase as it is shown by Durand [5].

Predicted air traffic growth is expected to double the number of flights over the next 20 years. Air Traffic Management will therefore have to absorb this additional burden and to increase airspace capacity, while ensuring at least equivalent standards of safety and interoperability. The European project SESAR (Single European Sky ATM Research) [3] was initiated to propose solutions to address this increasing demand problem. A major new concept of SESAR is the introduction of 4D trajectory planning. 4D trajectory planning consists of exploiting the possibilities of the flight management systems (FMS, system in charge of the aircraft navigation and controlling the auto-pilot) to ensure that a given aircraft is at a given position at a specified time. In this future framework, general curve trajectories will be designed. For each flight, a reference trajectory, called *Reference Business Trajectory* (RBT), is computed by the airline operating center and sent to aircraft in order to be managed by the FMS. In such a case, air traffic controller and pilots simply monitor trajectories computed by the planner.

Aircraft trajectory planning concerns the generation of a trajectory from start to goal taking into account several objectives, such as minimizing path distance, motion time or fuel, while avoiding obstacles in the environment and satisfying aircraft dynamics.

The problem of path planning in general deals with finding paths connecting different locations in an environment (e.g. a network, a graph, or a geometric space). Depending on the specific applications, the desired paths may need to sat-

isfy constraints (e.g. obstacle avoiding) and to optimize combinations of criteria (e.g. distance metrics and cost functions). The path planning problem can be formulated as an optimization problem that involves computing a collision-free path between two locations. This type of path planning is used in a large number of applications such as robotics, manufacturing, assembly, transportation, etc.

In this paper, two classes of path planning in *aircraft traffic management* (ATM) will be addressed. The first class considers static obstacles and is associated with the so-called *pre-tactical phase*. In pre-tactical phase, aircraft trajectories are altered to reduce congestion of airspace. Some aircraft trajectories are recomputed to avoid congestion areas. Such pre-tactical planning is also used to avoid bad-weather areas. Congestion areas and weather events may be considered as known for a given aircraft, one or two hours in advance. In our experiments, such areas will be considered fully "static" but our algorithm is capable to address dynamical obstacle. For this problem, the shapes and the locations of obstacles are assumed to be known. We are then searching for an aircraft trajectory that avoids obstacles while minimizing distance.

The second class of path planning studied here is dedicated to the *tactical phase*. As mentioned earlier, airways crossing points at beacons may induce risks of collision between aircraft, called *conflicts*. In the case of a potential conflict, aircraft trajectories are adjusted 3 to 7 minutes before its expected occurrence by air traffic controllers. This process is called tactical planning. In the second class of path planning, aircraft involved in a conflict are considered as "dynamic" obstacles in regards to one another. In this case, the goal is to identify a robust conflict-free trajectory for each aircraft.

For both classes of problems, the synthesized trajectories should be smooth in order to avoid sharp turns. The roll angle has to be bounded for passenger comfort. Furthermore, in order to fly, the aircraft speed has to stay within a bounded interval. The lower bound is imposed by the stall (wing do not produce any lift), and the upper bound is Mach 1 (destruction of the aircraft). All trajectories produced by planners must meet these speed constraints.

In the tactical phase, conflicts between aircraft are mainly solved by heading changes using standardized maneuvers:

- *turning points*: changing an aircraft heading and then bringing it back on its initial trajectory;
- *offsets*: inducing a lateral shift from the initial trajectory.

These maneuvers are elaborated by the air traffic controller and sent to pilots

by radio (vocal communication). When pilots are charged with controlling aircraft trajectories, it is impossible to resolve conflicts by sending aircraft a general curved trajectory for which a continuous heading change would be needed, due to the voice communication limitations. This last point limits the set of trajectories that can be considered for solving conflicts. However, in the future framework of SESAR, communications will be ensured by the system wide information management (SWIM). The SWIM architecture will be based on Internet protocols (data links between airborne and ground equipments). In this context, general curve trajectories can be shared between aircraft and ground.

In order to address trajectory planning problems, we propose an algorithm inspired by nature. Natural paradigms are at the basis of several optimization algorithms. Through billions of years, nature has found solutions to all the "problems" met. We can thus learn from the success of problem-solving by nature to develop nature-inspired heuristic algorithms. Classical nature-inspired optimization algorithms use two major components which are intensification (selection of the best solutions) and diversification (e.g. randomization). Intensification ensures that the solutions will converge to optimality, while diversification avoids the solutions from being trapped at local optima and at the same time, it broadens the range of proposed solutions. The best-known methods of this class are genetic algorithms [22], simulated annealing [18], ant colony optimization [4], bee algorithms [25] and particle swarm optimization [16].

Nature minimizes energy. In geometric optics, light behavior is modeled using rays. Light emitted from a point is assumed to travel along such a ray through space. In an effort to explain the motion through space taken by rays as they pass through various media, Fermat (1601-1665) developed his *principle of least action* [13]:

The path of a light ray connecting two points is the one for which the time of transit, not the length, is a minimum.

We can make several observations as a result of Fermat's principle :

- In a homogeneous medium, light rays are rectilinear. That is, within any medium where the index of refraction is constant, light travels in a straight line.
- In an inhomogeneous medium, light rays follow smooth geodesic curves with minimum transit time.

Light therefore tends to avoid high index areas where rays are slowed down. Light reaches lowest speed for the highest encountered index.

Based on this principle of least action, we introduce an optimal path planning algorithm which computes smooth geodesic trajectories in environments with static or dynamic obstacles. This algorithm mimics light propagation between a starting point towards a destination point, with obstacles modeled by high-index areas. By controlling the index landscape, it is possible the ensure that the computed trajectories meet the speed constraints and remain at a specified minimum distance from obstacles. Congestion and the *protection zone* (volume surrounding the aircraft where no other aircraft may enter) of other aircraft will be modeled as high-index areas. Our *light propagation algorithm* (LPA) is designed from a particular aircraft point of view. It is assumed that the aircraft knows the surrounding aircraft trajectories (the set of trajectories of the other aircraft is a given input of the algorithm).

The paper is organized as follows. Section 2 presents previous related works. Section 3 introduces the light propagation model and gives some theoretical clues linked to the synthesized trajectories. In Section 4, we describe our light Propagation algorithm, LPA, that compute approximate geodesics. In Section 5, we report numerical experiments on ATM problems. We show that LPA can solve 99% of the conflicts in the French airspace for a real traffic day (8000 trajectories).

0.2 Previous works

In robotics, motion planning has been treated as a *kinematic* problem, i.e. determining a path that avoids obstacles without concern to robot speeds. This was first extensively addressed for articulated robots by transforming the problem into the *configuration space*, in which the robot reduces to a point and the obstacles map into *C-space* obstacles [20, 21]. Using only obstacle-free paths computed using robot kinematics, may be dynamically infeasible even at moderate speed, causing the robot to deviate from the kinematic path due to its dynamics and limited actuator efforts. This gave rise to *dynamic motion planning*, which produces a trajectory in the *state space*, rather than just a path in the configuration space. Planning in the state space, while computationally more extensive, allows one to minimize a dynamic cost function, such as time or energy [30, 31, 32].

We distinguish between motion planning in static, and in dynamic environments. In static environments, the obstacles are fixed, and the robot is the only moving object. In dynamic environments, both the robot and the obstacles move.

Motion planning in dynamic environments was originally addressed by adding the time dimension to the robot's configuration space, assuming that the trajectories of the obstacles are given [8, 9, 26].

Another approach to dynamic motion planning is to decompose the problem into two sub problems: path planning and velocity planning. This method first computes a feasible path among the static obstacles in the spatial dimension. Then, the intersection of the moving obstacle with the path is represented as a forbidden region in the temporal dimension. The velocity along the path is chosen to avoid the forbidden regions [10, 11, 15].

Different approaches for path planning rely on extension of *visibility graphs* which are graphs of intervisible locations, typically for a set of points and obstacles in the Euclidean plane. Each node in the graph corresponds to a point location, and each edge represents a visible connection between them. In other words, if the line segment connecting two locations does not pass through any obstacle, an edge is drawn between them in the graph [12].

None of the previous trajectory planning methods considers the non-linear robot dynamics, except for the configuration space method. However, the latter does not take into account speed constraint. Furthermore, none of these approaches produces time optimal motions.

Several optimization methods have been proposed to resolve conflicts in air traffic [19]. The aim of these methods is to find optimal conflict-free 4D trajectories that reach the destination point. Such trajectories optimize a cost function which typically depends on the travel duration and on the *cost index* (the ratio of the time-related cost of airplane operations and the cost of fuel). The value of the cost index reflects the relative effects of fuel cost on overall trip cost, as compared with time-related direct operating costs. Methods for addressing conflict resolution problems are categorized into deterministic approaches and stochastic methods.

Genetic algorithms [7, 22] are stochastic methods that consist in generating a new "population" of aircraft trajectories from an initial 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 solution space is a set of finite maneuvers, which contains straight segments, turning points and offsets. These maneuvers are commonly used by air traffic controllers. Genetic algorithms generate trajectories with feasible operational maneuvers and with velocities within bounded ranges. They can reach an asymptotically optimal solution. However, for a given computing time, a feasible (conflict-free) solution is not guaranteed to be reached.

Navigation based approach [27, 2, 28, 29] is a deterministic method based on potential field for which aircraft are modeled by negative charges moving towards its destination modeled by high positive charge. Furthermore, they are formulated to avoid local traps with zero velocity which is a strong limitation for our aircraft conflict resolution problem. This produces a trajectory which connects the departure point to 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 ATM constraints, such as bounded speeds, smooth trajectories and time constraints. Besides, they may yield large deviations from the RBT.

For instance, genetic algorithms and the navigation-function based approach have been intensively studied, but none of them provides a completely satisfactory solution to the problem.

0.3 Light-propagation model

In this section, we describe the aircraft trajectory planning problem we are addressing, we present the light-propagation model and, finally, we discuss some theoretical aspects of our approach.

0.3.1 Problem statement

We seek to find for a mobile object, the shortest path between two points in $\mathbf{R}^n$, taking into account a given metric (time, distance, ...). The path is subject to two types of constraints plus the requirement that the object's velocity remains within a given bounded interval and given initial/final conditions (entrance, exit positions, head and speed). Furthermore, the curvature of this path is bounded in order to produce a smooth trajectory (which is critical for our problem for which aircraft heading change rates have to be bounded too).

The first type of constraint is hard; paths must not pass through obstacles represented by prespecified subsets of $\mathbf{R}^n$ (barriers). The second type of constraint is soft; the associated prespecified subsets of $\mathbf{R}^n$ should be avoided, but the path may go through them at the expense of a penalty term in the objective function. Another way to interpret these constraints is to consider that trajectories are longer

in these subsets, according to a metric used to compute the path's length. In both cases such constraints may either be static or dynamic.

For a given area, such hard and soft constraints are modeled by a "metric landscape" C. This metric landscape is a function that maps $\mathbf{R}^n$ to a cost value in $[0, +\infty[$. This metric can also depends on the time dimension in the case of dynamic constraints. In this case it is a function that maps $\mathbf{R}^n \times time$ to a cost value in $[0, +\infty[$. The shape of such a landscape enables the inclusion of soft and hard constraints. For example, sharp peaks represent hard constraints (obstacles) and smooth hills model soft constraints.

Let $\vec{\gamma}(\vec{s}, \vec{d}, u)$ be a mobile trajectory starting at point $\vec{s}$ and reaching the destination $\vec{d}$, where u is the curvilinear abscissa with

$$\vec{\gamma}(\vec{s}, \vec{d}, u_s) = \vec{s} \text{ and } \vec{\gamma}(\vec{s}, \vec{d}, u_d) = \vec{d}, \tag{1}$$

where u_s and u_d are the curvilinear abscissa of $\vec{s}$ and $\vec{d}$. In our model the travel time of the trajectory, t, is given by:

$$t(u) = t_s + u \times (t_d - t_s), \tag{2}$$

where t_s is the starting time at point $\vec{s}$ and t_d the time of arrival at the destination $\vec{d}$. The associated speed constraints to $\vec{\gamma}$ are modeled as:

$$\|\frac{\partial \vec{\gamma}(\vec{s}, \vec{d}, t(u))}{\partial t}\| \in]V_{min}, V_{max}[, \ \forall u \in [u_s, u_d], \tag{3}$$

where, for our problem, $\vec{V}_{min}$ is the stall limit and $\vec{V}_{max}$ is Mach 1 (at which the aircraft becomes unsafe). The smoothness constraint involves to the curvature, $K(\vec{\gamma}, u)$, of the trajectory at a given curvilinear abscissa u, and is defined by:

$$K(\vec{\gamma}, u) = \frac{\|\vec{\gamma}'(u) \wedge \vec{\gamma}''(u)\|}{\|\vec{\gamma}'(u)\|^3}, \tag{4}$$

where $\wedge$ denotes the cross product. Finally for a given trajectory $\vec{\gamma}$, the curvature is bounded above by some given K_{max}:

$$|K(\vec{\gamma}, u)| \leq K_{max} \ \forall u \in [u_s, u_d]. \tag{5}$$

We are therefore looking for a trajectory $\vec{\gamma}$ satisfying constraints (2), (3), (4) and (5), while minimizing the objective function:

$$f(\vec{\gamma}) = \int_{u=u_s}^{u=u_d} c(\vec{\gamma}(\vec{s}, \vec{d}, u))du, \tag{6}$$

where $c(\vec{\gamma}(\vec{s}, \vec{d}, u))$ is the cost of traveling the elementary space quantum du of the trajectory $\vec{\gamma}$ according to the metric c.

0.3.2 Model

We introduce a refractive index map which will be used to compute the metric landscape. This index map, I, is a function that maps $\mathbf{R}^n$ or $\mathbf{R}^n \times time$ (depending on the static or dynamic case) to an index value in $[1, +\infty[$.

The phenomenon of light propagation appears to be particularly well suited to model the search for a solution to the path planning problem.

Indeed, one can model the penalized areas (congestion areas or weather events) that should be avoided with zones of high refractive indices. The refractive index of a substance is a measure of the speed of light in that substance. It is expressed as the ratio of the speed of light in vacuum relative to that in the considered medium. One can then assign a refractive index proportional to the level of penalization.

The same concept can be used to forbid trajectories from entering the barrier areas by assigning infinite refractive indices to these subsets. In practice, we shall be content with assigning refractive indices to barriers that are much larger when compared to the rest of the index map. Areas where there are no constraints are assigned a refractive index equal to 1, which corresponds to the index of the vacuum, where light moves with maximum speed.

For each entry point to airspace, let us introduce a light source at the departure point emitting light beams in every direction. The path followed by the first light beam (which will be the synthesized aircraft trajectory) that reaches the destination point is the shortest path that we seek. Indeed, according to Fermat's principle, light follows the shortest-time path. This can also be interpreted mathematically as light follows geodesics, i.e. it uses a shortest path (if it exists) between two points in a space provided with a metric which is the path travel time. Light seeks to avoid areas with high indices (our soft constraints) because it is slowed down there, and it cannot pass through infinite-index areas (our hard constraints).

Another argument for modeling our aircraft path planning problem with light propagation is that, aside from the fact it fits well with our objective function and our penalty and barrier constraints, it also satisfies our speed-interval constraints. Indeed, light velocity being finite and depending on the medium's refractive index, we can guarantee solutions with speed remaining within a given interval simply by requiring media index values to remain within an appropriate range.

0.4 Computing geodesics

As noted above, the geodesic trajectory followed by light represents a potential solution to our path planning problem (the searched trajectory is the shortest in time). In this section, we concentrate on algorithms to compute geodesics. We first review the state-of-the-art methods from the literature. Then, we propose a practical branch-and-bound heuristic algorithm denoted LPA.

0.4.1 Triangle Mesh Algorithm

Triangle mesh algorithms look for approximate geodesics on a triangular mesh. First proposed by Kimmel and Sethian [17], they were later improved by Novotni [23] and enhanced by Tang [33]. These algorithms use front propagation from the departure point. It is similar to Dijkstra's algorithm for finding a shortest path in a graph.

In order to use the triangle mesh method, one must first create a mesh on a surface embedded in $\mathbf{R}^3$. However, it is not dedicated to find a geodesic on the whole $\mathbf{R}^3$ space, which would require building a full tridimensional mesh. A disadvantage of triangle mesh algorithms is that they cannot handle mobiles with variable speed.

0.4.2 Geodesic computation by light propagation algorithm

Let us assume for now that we numerically seek the path followed by light between two points in $\mathbf{R}^n$ space, the starting point, $\vec{s}$, and the destination point, $\vec{d}$, provided with a refractive index map. In the sequel of the paper, we shall mostly consider the special cases where $n = 2$ or $n = 3$, in view of our ATM application. First, we introduce a light source at the departure point. Then, we simulate the light propagation from this source using the wave propagation theory of light proposed by the Dutch scientist Christiaan Huygens in 1678. The Huygens principle [14] can be stated as follows: *any point on a wavefront can be considered as the source of tiny wavelets that propagate forward at the same speed as the wave, and the new wavefront is the envelope of all the wavelets* (that is to say, the tangent to these wavelets), as shown in Figure 1.

We propose here a wavefront propagation algorithm in $\mathbf{R}^n$ that we call light propagation algorithm, LPA. The light wavefront is issued from the starting point and is discretized in space. It is propagated in time dimension using a time step dt.

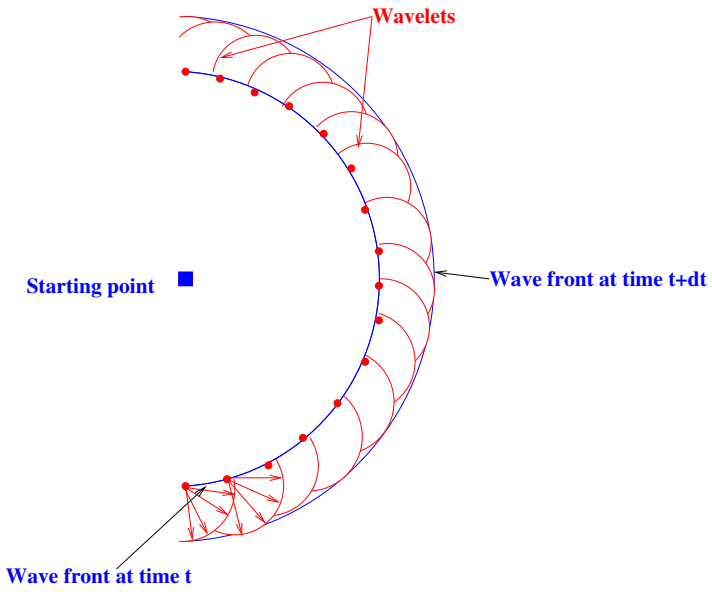


Figure 1: Huygens principle used to determine the wave front at time $t + dt$ from the wave front at time t

We implement LPA within a *branch-and-bound algorithm* (B&B) [1], a classical framework for solving discrete optimization problems. A B&B algorithm represents the set of all possible solutions by the root of an enumeration tree. Constructive procedures to obtain lower and upper bounds for the optimal value of our objective function (trajectory travel time) are first applied to the root. If these two bounds are equals, then the optimal solution is found, and the algorithm stopped. Otherwise, the solution set is partitioned into several sub problems (new children nodes). This process is called *branching*. The method is then applied recursively on the corresponding sub problems, generating a tree. If an optimal solution is found for a sub problem then it is feasible but not necessarily optimal for the original problem. Based on the bounds, feasible solutions are used to eliminate sets of partial solutions, reducing the size of the search tree (cutting branches). The search goes on until all the nodes are explored or eliminated.

For the implementation of B&B in the context of our light propagation model, we propose to compute an approximate lower bound for a given node by the sum-

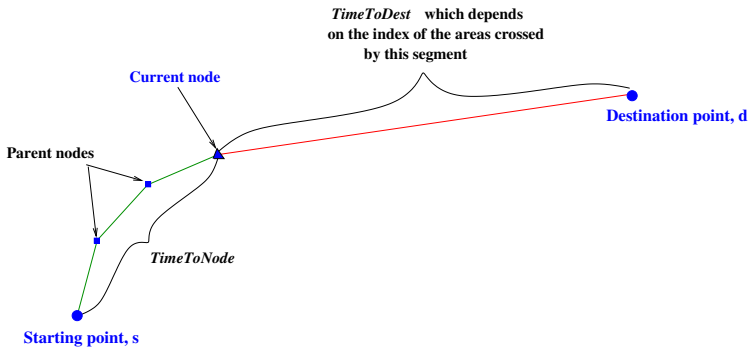


Figure 2: The approximate lower bound computation for the current node. It is computed by adding the time required to reach the current node and the time to reach the destination point from this node using the direct straight line trajectory. Those times are computed taking into account the encountered index value.

mation of two terms (see Figure 2):

$$approxLB := TimeToNode + TimeToDest,$$

where the first term, "*TimeToNode*" is the time that is required to reach the current node from the origin, and the second term, "*TimeToDest*", represents the remaining time to reach the destination. This duration is a weighted sum of two terms which are "*integralTime*" and "*maxSpeedTime*". The first term, integralTime, is the time to reach destination considering the refractive index along the direct route, denoted by γ_{direct} and the speed used for this computation depends on the index encountered along γ_{direct} ($speed = \frac{\text{maximum speed}}{index}$ where "*maximum speed*" is the speed assuming unit refractive index). The second term, maxSpeedTime, is the time needed to reach the destination along γ_{direct} when traveled at maximum Speed. This is given by Formula (7).

$$TimeToDest := \alpha * integralTime + (1-\alpha) * maxSpeedTime \qquad (7)$$

$$= \alpha \int_{u_{node}}^{u_d} C(I(\vec{\gamma_{direct}}(u)))du \; + \; (1-\alpha)\frac{\|\vec{\gamma}_{direct}(u_d) - \vec{\gamma}_{direct}(u_{node})\|}{MaxSpeed}$$

where α is a weighting parameter set by the user; u is the curvilinear abscissa; u_{node} and u_d are curvilinear abscissa of the current node and of the destination; I

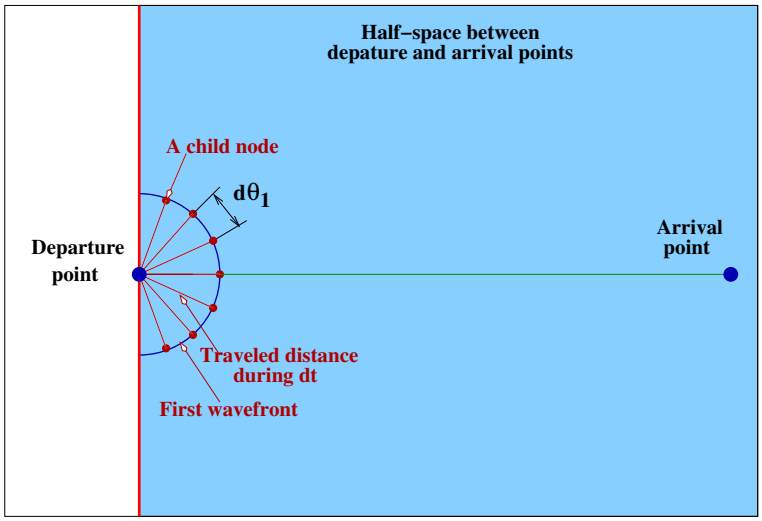


Figure 3: Half-space discretization with a time step dt and an angle step $d\theta_1$.

is the index map; and C the metric landscape that represents the time needed to travel an elementary space quantum with index I.

The wavefront propagation will be done by means of the branching process of the B&B algorithm. The method projects light beams in straight lines from the current node. The emission of light beams is restricted in the direction in the half space between the current node and the arrival point (standard aircraft operations does not permit travel in the opposite direction, except in the neighborhood of the origin or destination airports). These beams are launched in all such directions according to beam-launching angles $\theta_1, \theta_2 ... \theta_{n-1}$ in the spherical coordinate basis of $\mathbf{R}^n$, and discretization angle steps: $d\theta_i$, $i = 1, 2, .., n-1$ respectively. Each beam propagates in one of these resulting directions with a velocity that depends on the refractive index of the medium through which it passes, reaching a child node of the current node after a time step dt. These algorithmic parameters $d\theta_i$ and dt are set by the user. All nodes that have the same depth in the resulting tree will represent the same wavefront (see Figure 3 for an example with $n = 2$).

Next, LPA browses the search tree to find a minimal-time trajectory (an approximate geodesic). This can be done in different ways. We choose a strategy whose priority is to find quickly a feasible solution with *Depth-First Search* (DFS). It consists in choosing a node for which children have not yet been generated, with deepest level in the search tree. DFS is then combined with the selection

strategy of choosing the node that has the best lower bound among the nodes at the same level in the search tree. In other words, we combine DFS as the overall principle with *Best-First Search* (BFS) as a secondary selection criterion.

Figures 3, 4 and 5 illustrate the algorithm operation in $\mathbf{R}^2$. Figure 3 displays an initial wavefront from the starting point, s. Figure 4 shows wavefront deformation for the current node N. Finally, figure 5 shows the trajectory produced by the algorithm.

In order to describe our algorithm, we assume that the user has set a distance-from-destination tolerance $\epsilon > 0$ (see Figure 5). Moreover, we shall need a procedure "*LaunchRays(*N*)*" for a node N. This procedure is used as the branching process of the B&B algorithm:

Procedure LaunchRays(N)

i. Discretize the half-space between node N and the destination point with a time step dt and angle steps $d\theta_i$, $i = 1, 2, ..., n-1$.

ii. Determine new child nodes and their values using the following rule:

For any light beam, if it goes into a region with index n, its velocity inside this region is $v = \frac{c}{n}$, where c is the maximum speed. For any child node, its associated value is the approximate lower bound described above.

iii. Remove node N from the tree and add its children.

Remark that when this procedure will be used for aircraft trajectories planning in our result section, we shall replace maximum speed (above) with the speed of aircraft.

The main steps of LPA are as follows:

1. Set TrajSolution := $\emptyset$. Set upperBound := ∞.

2. *LaunchRays(s)* (see Figure 3).

3. While there is still unexplored nodes in the tree, choose a node N according to DFS and then BFS as described above. Then :

If distance(N, $\vec{d}$)$\leq \epsilon$ and value of node N $\leq$ upperBound then

TrajSolution := Set of ascendant points that lead to N

and upperBound:= value of node N (see Figure 5).

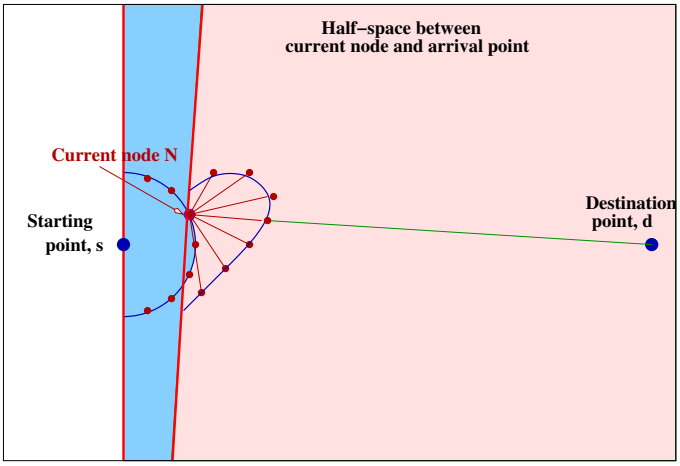


Figure 4: Launching rays from current node N.

1. Else

LaunchRays(N) (see Figure 4).

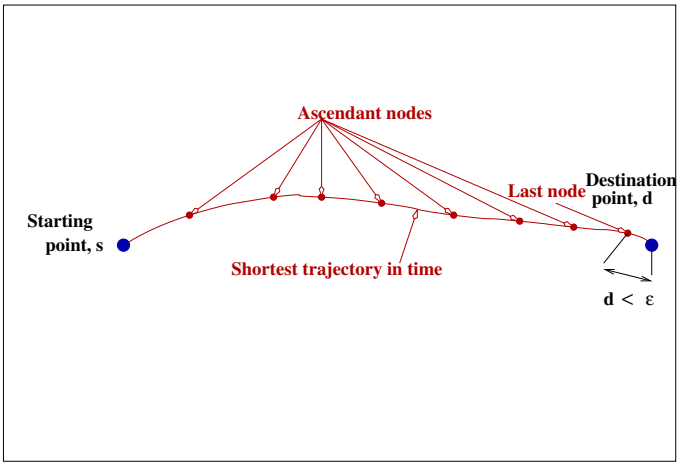


Figure 5: A shortest trajectory between the departure and the arrival points.

0.4.3 Light propagation algorithm with dynamic obstacles

To address the case of dynamic obstacles, which is crucial in ATM applications, we adapt the algorithm to take into account the extra additional dimension of time. This dimension is different from spatial dimensions as propagation in this coordinate is one way. In the pure spatial case, one has to connect the starting point, $\vec{s}$, to the destination point, $\vec{d}$, through a trajectory $\vec{\gamma}$. A simple manner in which to extend LPA to take into account the extra time dimension is to add the time coordinate to the points $\vec{s}$ and $\vec{d}$, and to find a 4D path between them. By forcing the light beams to propagate one way in the time dimension, the previous algorithm may be directly adapted in this new coordinate system. However, resulting trajectories may then violate aircraft speed constraints. For instance, if the original path $\vec{\gamma}_{old}$ between two points ($\vec{s}$ and $\vec{d}$) is a straight line with associated dates t_s and t_d, the new trajectory produced by the straight forward extension of the algorithm may be longer. In order to reach the time target t_d at destination d, the synthesized trajectory may induce a speed profile that is incompatible with flight constraints. In some other situations, the new path may be shorter in the space dimension, inducing an excessive speed reduction. To avoid these pitfalls, we propose an adaptation of LPA that replaces the target in the time dimension with a *time segment target* at destination, as illustrated in Figure 6. This modification ensures the feasibility of the produced trajectories from the speed constraint point of view.

0.5 Numerical Results

In this section, we present numerical experiments for several ATM problems.

First, we test LPA on the pre-tactical phase problem, with static obstacles for which high indices model congestion areas and weather events have to be avoided. As mentioned before, LPA can deal with dynamical obstacles but we implement it with static obstacles in the first test for simplicity. Secondly, we report results with LPA in the tactical phase, with several aircraft involved in a conflict. Finally, we consider a real-world problem involving a day of traffic over the French airspace.

In each of our tests, we set the sampling angle $d\theta_1$ to $\frac{\Pi}{36}$. The weighting coefficient α is set to 0.9. All experiments are performed on a 2.33 GHz machine running under Mandriva Linux operating system with 1024 MB of RAM. LPA is programmed in java.

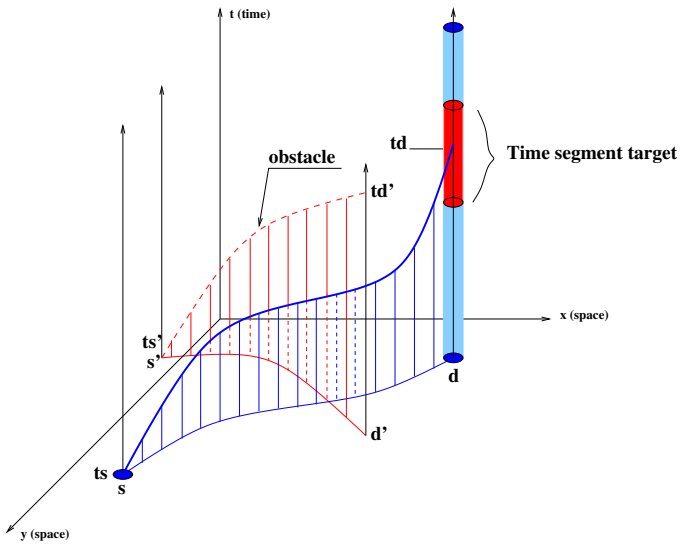


Figure 6: Two trajectories in a 2D+time coordinate system. The dashed trajectory represents a moving obstacle (another aircraft going from $\vec{s'}$ to $\vec{d'}$ and arriving at time t'_d) and the plain line represents the trajectory controlled by the LPA and reaching a time target segment containing t_d.

0.5.1 Trajectory planning with static obstacles

In order to validate our algorithm, we first test LPA in $\mathbf{R}^2$ space (no time involved) to which we associate a static refractive index map. The goal is to find geodesics between two given points in $\mathbf{R}^2$.

We use a coordinate system that is scaled according to 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 40×40 standard horizontal separations.

We test our algorithm by using an artificial index map given by the following formula: $I(x,y) = max(1, \sum_{i=0}^{4} e^{-\frac{((x-a_i)^2+(y-b_i)^2)}{k}})$ where a_i, b_i and k are given. By setting different coefficient values for a_i, b_i and k, we have created five different maps, as displayed in Table 1. Figure 7 displays trajectories produced by LPA for various obstacles and with maps defined by the parameter values of Table 1. High values, corresponding to region of congestion, are represented in dark areas surrounded by white.

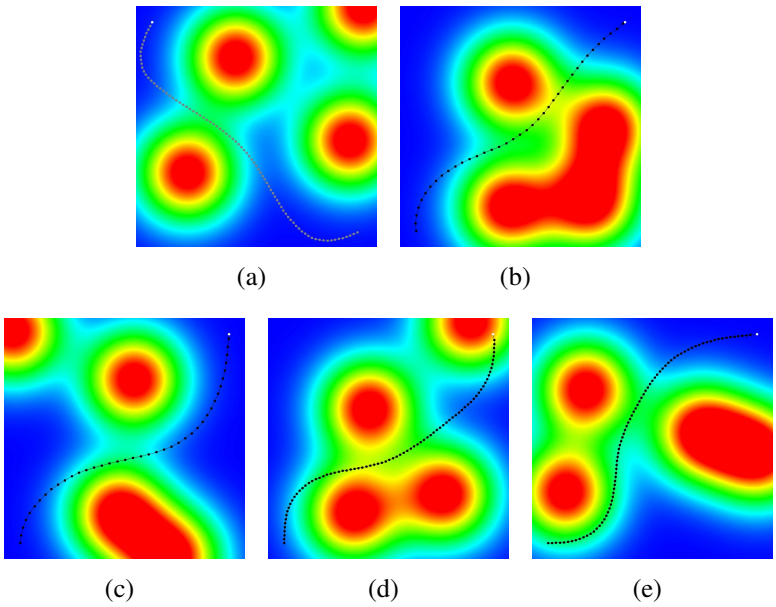


Figure 7: Resulting trajectories in $\mathbf{R}^2$ space (flying from bottom to top).

	a_1	b_1	a_2	b_2	a_3	b_3	a_4	b_4	k
(a)	10	10	18	130	34	18	36	44	0.1
(b)	20	8	20	28	30	9.2	32	18	0.1
(c)	20	10	0	38	26	0	20.8	28	0.1
(d)	16	8	18	22	28	10	32	40	0.1
(e)	8	10	10	20	36	18	26	19.6	0.1

Table 1: Index function coefficients for each of the five instances illustrated in Figure 7.

The five solution trajectories displayed in Figure 7 are found by LPA in less than 5 seconds of CPU time. Each trajectory produced by LPA is a geodesic approximation that avoids high index areas and passes through low-value "valleys". Moreover, trajectories produced by LPA are guaranteed to have velocity above the predetermined speed lower bound. This is crucial in our ATM applications. Finally, these trajectories are, by construction, sequences of segments and arcs, which can be managed by the FMS.

Although the test problems we consider in this subsection are academic, LPA can be applied to real-world aircraft trajectories in pre-tactical planning to avoid

convective-weather or congested areas that are considered as static obstacles aircraft try to avoid (but can cross with high penalty).

0.5.2 Application to aircraft conflict resolution

In this subsection, we consider the tactical phase for the aircraft conflict resolution problem involving *several* aircraft. We consider as given input a set of aircraft trajectories in conflict and for which new conflict-free trajectories have to be designed. For each aircraft, a, involved in the conflict zone, a new trajectory with approximate minimum deviation is generated, between the aircraft entry point, $\vec{s_a}$, and its exit point, $\vec{d_a}$.

To address this conflict resolution problem, aircraft are sequentially resolved using LPA. We assign a trajectory to the first aircraft disregarding the other aircraft (without considering any constraints). Then, LPA looks for a trajectory for the subsequent aircraft by considering the trajectory of the first aircraft as a constraint, and so on, up to the n^{th} aircraft which considers the $n-1$ previous aircraft trajectories as constraints. The trajectory assigned to an aircraft, a, in each resolution step must connect two points in 3D space (the aircraft starting point $\vec{s_a}$ and its destination point $\vec{d_a}$) and must avoid other aircraft trajectories which are considered as fixed constraints (known data). In our case, the aircraft ordering is chosen at random. In practice, some operational criteria may also be used in order to select a specific sequence (for instance: first-come first-served rule, some aircraft may have higher priority, trajectory length, etc.).

More specifically, we adapt LPA to the aircraft conflict resolution problem as follows :

- Working in 4D ($\mathbf{R}^3$+time), the propagation in the time dimension is done exclusively in one direction (from past towards future) with a time step dt. The time step is adjusted by the user in order to ensure proper conflict detection. Depending on the speed of the aircraft, we choose to fix it to the duration needed to fly a half horizontal standard separation distance ($2.5Nm$). The time segment target is set at $[t_s, t_d + 0.1 \times (t_d - t_s)]$, where t_s and t_d are respectively the associated dates of the stating and destination points respectively.

- In order to resolve conflicts in the cruise phase, only lateral deviations are allowed in the algorithm in order to keep the optimal vertical profile of aircraft. Conflict avoidance is then accomplished by heading changes.

This is in accordance with common ATM practice: level changes are much more expensive from the fuel consumption point of view; moreover, vertical change is less comfortable for passengers. Propagation in the vertical dimension is then adjusted in order to match the original aircraft vertical profile. Hence, the user only needs to provide *one* discretization angle, $d\theta_1$, as if we were working in $\mathbf{R}^2$+time.

- In the branching phase (Figure 4), instead of launching light beams in the half space between the current node and the destination point, we choose to launch them into a cone pointed at the current node and directed towards the arrival point. The angle of this cone is set to $\frac{\pi}{6}$ in our tests. This prevents undesirable sudden heading changes higher than $\frac{\pi}{6}$ and allows the aircraft trajectory to remain within a certain envelope around the direct route (recall that one objective is to keep the trajectory solution as close as possible to the original route).

- Ray tracing velocity is no longer constant (equal to the maximum speed, c), but depends on the specific aircraft velocity profile. This velocity profile is an input of the problem that gives the speed of the aircraft at each time. When propagating rays, one easily obtains the speed of the aircraft at time value *TimeToNode* associated with the current search tree node.

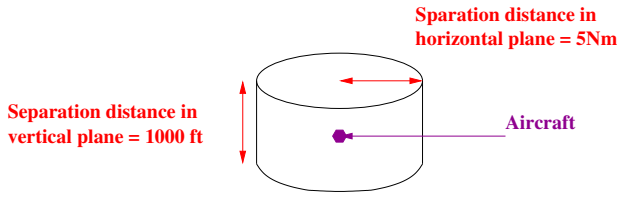


Figure 8: Aircraft protection zone cylinder in 3D

- At the n^{th} step, LPA synthesizes a trajectory for the n^{th} aircraft in $\mathbf{R}^3$+time space that must avoid 4D tubes representing the $n-1$ previous aircraft trajectories (already computed). The section of such a 4D tube at time t, is the aircraft protection zone (in $\mathbf{R}^3$). It is a cylinder whose basis is a disk of radius equal to the minimal separation distance between two aircraft in the horizontal dimension (5 Nm) and whose height is the minimal separation distance in the vertical dimension (1000 fts), as illustrated in Figure 8. To

guarantee conflict-free aircraft trajectories, LPA directly eliminate any ray that enters such a 4D tube within the B&B process.

The refractive index function associated with this second test problem must guarantee avoidance with the other aircraft 4D tubes. The index function, I, is set to a high constant value (I_{max}) inside these tubes and elsewhere it is set to the value 1. Let $\vec{Y_i}(t)$, $i = 1, 2, ..., n-1$ be the positions at time t of the $n-1$ aircraft already treated. The index function I is given by the following formula at any point $\vec{X}(t) \in \mathbf{R}^3$:

$$I(\vec{X}(t)) = \begin{cases} I_{max} & \text{if } \exists\, i \text{ such that } (d_v(\vec{X}(t), \vec{Y_i}(t)) \leq 1000\ ft) \text{ and } (d_h(\vec{X}(t), \vec{Y_i}(t)) \leq 5\ Nm) \\ 1 & \text{otherwise,} \end{cases}$$

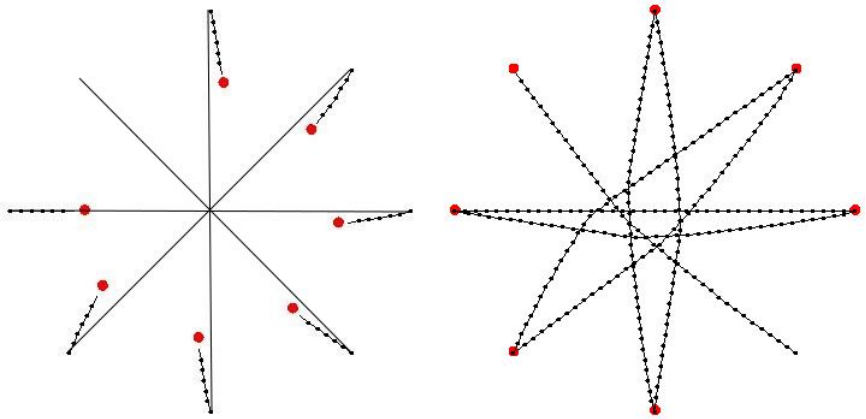
where, considering two points $\vec{A} = \begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix}$, $\vec{B} = \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} \in \mathbf{R}^3$, $d_v(\vec{A}, \vec{B}) = |b_z - a_z|$ (distance in the vertical plane) and $d_h = \sqrt{(b_x^2 - a_x^2) + (b_y^2 - a_y^2)}$ (distance in the horizontal plane).

We consider an artificial problem instance involving P aircraft which are initially located on a circle of radius 100 Nm, converging at the same speed (450 knots velocity) towards the center of the circle, at the same flight level (FL) 300 (a standard nominal altitude of an aircraft, in hundreds of feet). The algorithm is sequentially applied to each aircraft with $P = 7$ aircraft involved and $I_{max} = 2$. To detect conflicts, we use a 4D grid with a standard horizontal separation unit in the plane (x, y) and standard vertical separation unit in the vertical plane.

The set of resulting solution trajectories is found by LPA in less than 30s of CPU time. The solution result is a conflict-free situation where, as expected, the first aircraft does not deviate from its direct route, as displayed on Figure 9.

This problem is a classical benchmark in aircraft trajectory planning and other studies also find similar (roundabout-like) results for this academic problem, including Durand's approach [6] that uses genetic algorithms and modifies trajectories with offset maneuvers. The result obtained by Durand on a similar problem with $P = 6$ aircraft is shown on Figure 10. The first aircraft keeps a direct trajectory and all other aircraft make an offset to the left.

Similar results have also been found for a case involving $P = 5$ aircraft by Pallottino and Feron [24] with mixed integer programming. They modify the initial heading of aircraft by the minimal angle that permits to avoid conflicts.



(a) (b)

Figure 9: Conflict resolution with 7 aircraft converging towards the center of a circle. Straight line trajectories in (a) represent aircraft trajectories before conflict resolution. (b) shows the solution trajectories produced by the LPA.

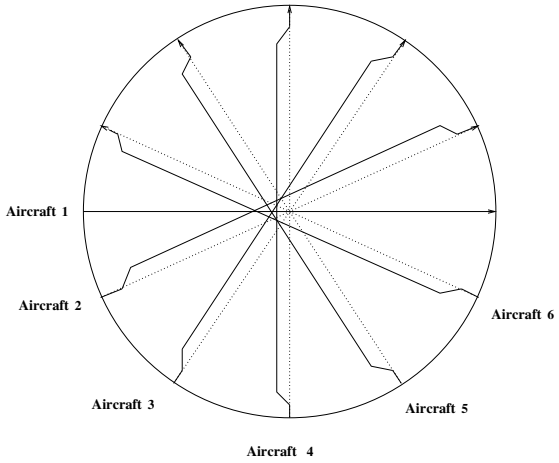
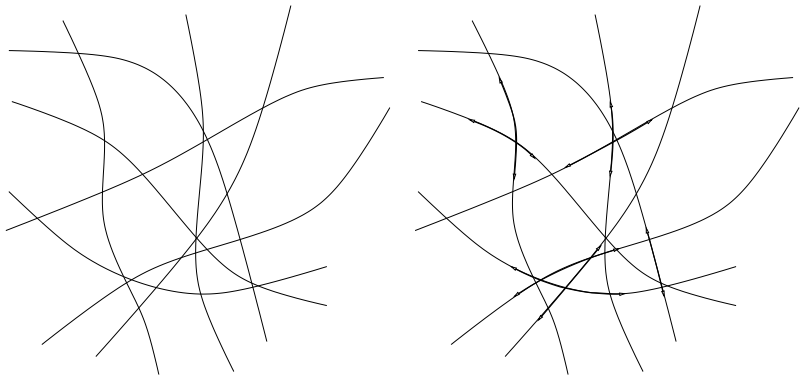


Figure 10: Conflict resolution by Durand (genetic algorithm and using offset maneuvers) for 6 aircraft.

The algorithm is applied iteratively to reach the original goal of each aircraft. The only difference between the results of Pallottino and Feron and ours results is that in their case every conflicting aircraft changes its heading (there is no privileged aircraft).

0.5.3 Application to real-world data

We consider here the same aircraft conflict resolution problem as in the previous subsection, but on a real-world problem. We test LPA on a real day of traffic for the French airspace. In order to simulate a day of traffic in the the French airspace, we use historical flight plans for a given day which represent the demand for aircraft traffic.



(a) (b)

Figure 11: A complete trajectory set (a) and trajectory segments relevant to a given time window (in bold) (b).

Flight plans are documents filled by pilots or by a flight dispatcher with the local civil aviation authority (e.g. Federal Aviation Administration in the USA) prior to departure. They generally include basic information such as departure and arrival points, the route, etc. The route is designed by a list of waypoints. These flight plans are then used as input for a traffic simulator named CATS (Complete Air Traffic Simulator), developed by the former French Civil Aviation Research Center (CENA). The core of the CATS system is an en-route traffic simulation engine used to produce sampled trajectories. It is based on a discrete, fixed-time slice execution model: the position and velocities of aircraft are computed at fixed time steps, usually every 5, 10 or 15 seconds. Aircraft performances are in tabulated form describing ground speed, vertical speed, and fuel burnt as a function of altitude, aircraft type and flight segment (cruise, climb or descent). These performances are extracted from the Eurocontrol aircraft data base BADA. The simulator computes the associated trajectories set for each flight plan of the input set (see Figure 11(a)). Each trajectory is a list of points sampled every 15 seconds.

In order to build our instance, we proceed iteratively using moving time windows. In our tests, time windows are set to 21 minutes. For each time window, we extract the relevant trajectory segments from the complete trajectory set (see Figure 11(b)). In order to detect conflicts, each segments is included in a box. The box is the smallest rectangular cuboid including the segment, enlarged by a safety buffer equal to half the separation norm (see Figure 12).

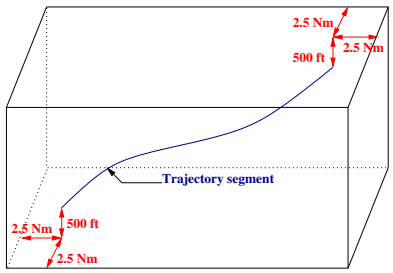


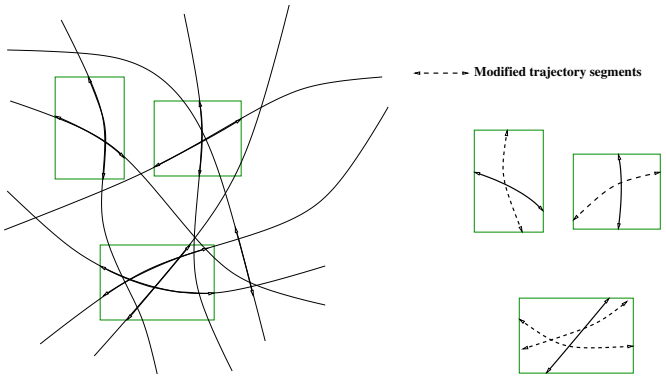
Figure 12: The box including a trajectory segment.

When boxes intersect, the associated trajectories are considered potentially in conflict. To detect actual conflicts, trajectory segments are embedded in a grid whose elements are of size $5Nm \times 5Nm \times 1000ft$. The trajectory segments that are actually in conflict are gathered together in a same including rectangular cuboid and are called *conflict clusters* (see Figure 13(a)). When modifying trajectory segments to resolve conflicts, we only allow new trajectory segments to be inside the including rectangular cuboid. Moreover, all other trajectories present in this rectangular cuboid (the ones that are not in conflict) are introduced into the cluster in order to be treated as constraints of the problem.

All clusters are then solved separately by LPA in order to produce conflict-free trajectories (see Figure 13(b)). The old trajectory segments are then replaced by the solutions produced by LPA. The remaining part of each trajectory (after the current time window) is then updated according to the time and speed shifts computed by LPA (see Figure 14(a)).

Then, aircraft are "allowed" to fly for a period corresponding to a fraction of the time window (one third, in our experiments) during which aircraft follow the new trajectories. Then, this process is repeated for the next time window (see Figure 14(b)).

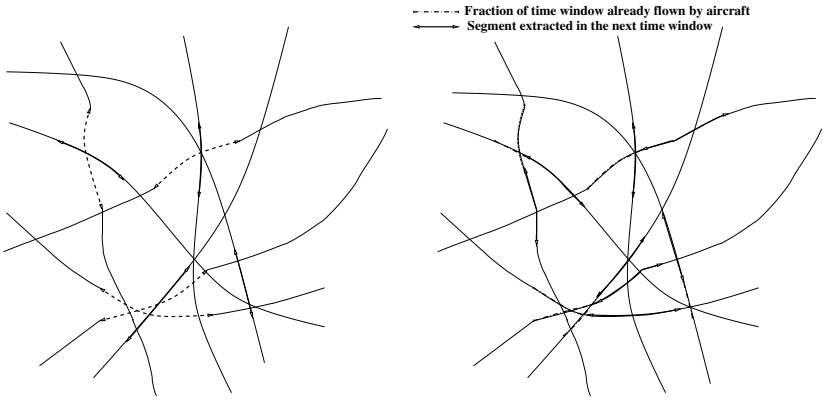
We now present results produced on a day of traffic (August 12, 2008) with about 8000 aircraft. The initial trajectories (before conflict resolution) induce a



(a) (b)

Figure 13: Conflict detection (a) and conflict resolution (b).

total number of clusters, with some aircraft in real conflict, equal to 3344. The algorithm nearly solves all conflicts, with only 28 situations for which conflict-free trajectories have not been found. However, these situations correspond to some aircraft being already in conflict at the beginning of the simulation, for instance at their starting point. Only 1501 trajectories have been modified to reach such a



(a) (b)

Figure 14: Trajectories update (a) and next iteration (b)

conflict-free planning. In many cases, the new computed trajectories are shorter than the initial ones (those that follow waypoints), due to the fact that LPA is searching for the shortest path trajectories and proposes direct routes when possible. On average, the length of the computed trajectories is shortened by 4.41 Nm

per aircraft, with a minimum of 51.9 Nm. On the other hand, some trajectories undergo length extension with a maximum of 9.86 Nm. The associated quantiles are displayed in Table 2:

Quantile	Distance in Nm
$\frac{1}{4}$ quantile	-4.94
$\frac{1}{2}$ quantile	-1.65
$\frac{3}{4}$ quantile	-0.07

Table 2: Trajectory length changes after conflict resolution by LPA.

The overall computation time is about 17 hours. We expect to reduce substantially this computation time by rewriting some parts of the software into C language. On a second step, for a given time window, the current clusters (which are independent) will be solved on several machines in parallel.

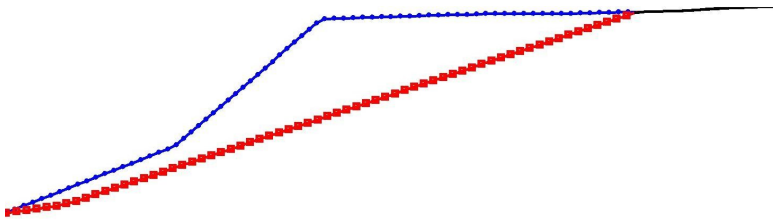


Figure 15: Example of conflict resolution yielding a shorter distance. The simple line represents the future trajectory. The line with circles represents the initial trajectory in the current time window and the line with squares represents the trajectory produced by LPA.

Some specific examples of resolutions are displayed in Figures 15, 16 and 17. The past and future parts of the trajectories are represented by simple lines. The current time window is represented by lines with squares or circles. The line with circles represents the initial trajectory (with conflict), and the line with squares is the solution produced by LPA (without conflict). The solution trajectories met the aircraft speed constraint (by following aircraft speed profile) and are smooth, as required by airliners (thanks to the heading-change constraint, when rays are launched from the current position). The synthesized trajectories are fully adapted to be followed by the FMS.

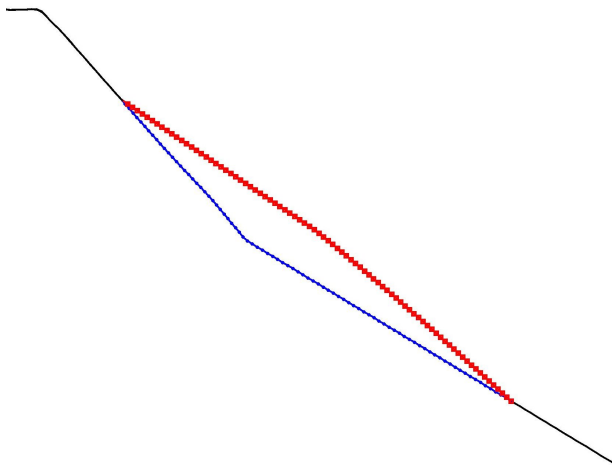


Figure 16: An example of conflict resolution without change in distance.

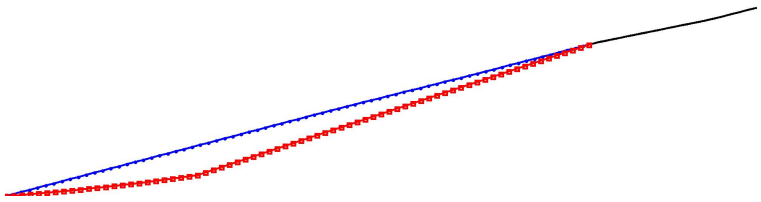


Figure 17: Conflict resolution yielding a longer distance.

0.6 Conclusion

Aircraft conflict resolution is a crucial issue for air traffic management (ATM). Much work and many efforts have been dedicated to address this problem. This is the key issue to increase the capacity of the future ATM system. Robot motion planning, navigation functions and some other previous related works cannot be used for such problems due to typical aircraft trajectory constraints.

Light rays follow smooth geodesic curves with minimum transit time for which the minimum speed is controlled by the maximum index encountered by the light ray. Based on this natural phenomenon, we have developed a new path planning algorithm, called LPA, which mimics the light propagation behavior. By control-

ling the index landscape, it is possible the ensure that the resulting trajectories meet the speed constraints and are at specified minimum distance from obstacles. We have presented two implementations of LPA in order to address pre-tactical and tactical aircraft path planing. In the first case, obstacles are considered static and the algorithm optimizes aircraft trajectories for congestion areas or weather-event avoidance. The second implementation of LPA that we introduced can deal with dynamical moving obstacles such as encountered aircraft. In this case, light rays propagate in a four dimensional space (3D + time) where propagation is restricted to one direction for the time dimension.

We applied successfully these two variants of LPA on three ATM problems. In the first case, LPA found minimum-time trajectories avoiding congestion areas (considered as static smooth constraints). In the second case, LPA solved a classical benchmark problem in the tactical phase for which several aircraft are involved in a conflict. Finally, LPA yielded very encouraging results on a real-world tactical phase ATM problem involving numerous aircraft in various conflict situations.

Current work involves improving the algorithm performances by computing cluster resolutions in parallel. Finally, in order to take into account uncertainties in aircraft positions, this algorithm will be extended to produce robust solutions. Our model, supports both temporal congestion and moving weather. In future work, one could apply our methodology to such situations. It would also be interesting to use our model with more general cost functions, replacing the traveled time (delays) criteria with, for instance, fuel burn or cost index (a compromise between the cost of fuel and the cost of time). Besides, future work could concentrate on solving the problem involving *several* coordinated aircraft in a more general manner. Instead of relying on sequentially applying our light propagation methodology to one aircraft at a time (the previous ones considered fixed), one could attempt at solving the problem *globally* (all aircraft moving simultaneously). This is a non trivial challenging issue.

Bibliography

[1] Balas, E., Toth, P.: Branch and bound methods in the traveling salesman problem. John Wiley & Sons (1985). Pp 361-401

[2] Chaloulos, G., Roussos, G., Lygeros, J., Kyriakopoulos, K.: Ground assisted conflict resolution in self-separation Airspace. In: AIAA Guidance, Navigation and Control Conference and Exhibit. Honolulu, Hawaii (2008)

[3] Consortium, S.: The european ATM master plan. Tech. Rep. 1, European Commission and EUROCONTROL (2009)

[4] Dorigo, M.: Optimization, learning and natural algorithms. Ph.D. thesis, Politecnico di Milano, Italie (1992)

[5] Durand, N.: Optimisation de trajectoires pour la résolution de conflits en route. Ph.D. thesis, INPT (1996)

[6] Durand, N.: Algorithmes génétiques et autres méthodes d'optimisation appliquées à la gestion de trafic aérien. Ph.D. thesis, Institut National Polytechnique de Toulouse (2004). Habilitation à diriger des recherche

[7] Durand, N., Alliot, J.: Optimal resolution of en-route conflict. In: Eurocontrol/FAA ATM Seminar. Eurocontraol/FAA (1997)

[8] Erdmann, M., Lozano-Perez, T.: On multiple moving objects. Algorithmica **2**, 477–521 (1987)

[9] Fujimura, K., Samet, H.: A hierarchical strategy for path planning among moving obstacles. IEEE Transactions on Robotics and Automation **5**(1), 61–69 (1989)

[10] Fujiruma, K.: Time-minimum routes in time-dependent networks. IEEE Transactions on Robotics and Automation **11**(3), 342–351 (1995)

[11] Fujiruma, K., Samet, H.: Time-minimal paths among moving obstacles. In: IEEE International Conference on Robotics and Automation. IEEE (1989)

[12] Fujiruma, K., Samet, H.: Motion planning paths among moving obstacles. In: IEEE International Conference on Robotics and Automation. IEEE (1990)

[13] Giancoli, D.C.: Physics for Scientists and Engineers with Modern Physics, second edn. Prentice hall (1989)

[14] Huygens, C.: Traité de la lumière. Leiden, Netherlands. Pieter van der Aa (1690)

[15] Kant, K., Zucker, S.: Towards efficient trajectory planning: The path-velocity decomposition. The International Journal of Robotic Research **5**(3), 72–89 (1986)

[16] Kennedy, J., Eberhart, R.: Swarm Intelligence. Morgan Kaufmann (2001)

[17] Kimmel, R., Sethian, J.: Computing geodesic paths on manifolds. In: Proceeding of Natianal Academy of Sciences of USA, pp. 8431–8435 (1998)

[18] Kirkpatrick, S., Gelatt, C., Vecchi, M.: Optimization by simulated annealing. Science **220** (1983)

[19] Kuchar, J., Yang, L.: A review of conflict detection and resolution modeling methods. IEEE Transactions on Intelligent Transportation Systems **1**(4), 179–189 (2000)

[20] Latombe, J.: Robot Motion Planning. Kluwer Academic Publisher (1991)

[21] Lozano-Perez, T., Wesley, M.: An algorithm for planning collision-free paths among polyhedral obstacles. Communication of the ACM **22**(10), 560–570 (1979)

[22] Michalewicz, Z.: Genetic algorithms + Data Structures = Evolution Programs. Springer-Verlag (1992)

[23] Novotni, M., Klein, R.: Computing geodesic distances on triangular meshs. In: Proceeding of the 10th International Conference in Central Europe on Computer Graphics Visualization and Computer Vision, pp. 341–347 (2002)

[24] Pallottino, L., Feron, E., Bicchi, A.: Conflict resolution problems for air traffic management systems solved with mixed integer programming. In: Transactions on Intelligent Transportation Systems. IEEE (2002)

[25] Pham, D., Ghanbarzadeh, A., Koc, E., Otri, S., Rahim, S., Zaidi, M.: The bees algorithm: A novel tool for complex optimisation problems. In: Proceedings of IPROMS Conference. IPROM (2006)

[26] Reif, J., Sharir, M.: Motion planning in the presence of moving obstacles. In: 25th IEEE Symposium of the Foundation of Computer Science. IEEE (1985)

[27] Roussos, G., Chaloulos, G., Kyriakopoulos, K., Lygeros, J.: Control of multiple non-holonomic air vehicles under wind uncertainty using model predictive control and decentralized navigation functions. In: IEEE Conference on Decision and Control. IEEE (2008)

[28] Roussos, G., Dimarogonas, V., Kyriakopoulos, K.: 3D navigation and collision avoidance for a nonholonomic aircraft-like vehicles. International Journal of Adaptive Control and Signal Processing (2009)

[29] Roussos, G., Kyriakopoulos, K.: Towards constant velocity navigation and collision avoidance for autonomous nonholonomic aircraft-like vehicles. In: Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. Proceedings of the 48th IEEE Conference on, pp. 5661 – 5666. Shanghai (2009)

[30] Shiller, Z.: Time-energy optimal control of articulated systems with geometric path constraints. ASME Journal of Dynamic Systems, Measurements and Control **118**(1), 139–143 (1996)

[31] Shiller, Z., Dubowsky, S.: Robot path planning with obstacles, actuators, gripper and payload constraints. The International Journal of Robotics Research **8**(6), 3–18 (1989)

[32] Shiller, Z., Gwo, R.: Dynamic motion planning of autonomous vehicles. IEEE Transactions of Robotics and Automation **7**(2), 241–249 (1991)

[33] Tang, J., Zhang, F., Zhang, M.: Fast approximate geodesic paths on triangle mesh. In: Proceeding of the International Journal of Automation and Computing, pp. 8–13 (2007)