FACES : a Free flight Autonomous and Coordinated Embarked Solver

Jean-Marc Alliot, Nicolas Durand, Géraud Granger

FACES: a Free flight Autonomous and Coordinated Embarked Solver

Jean-Marc Alliot **Nicolas Durand** **Géraud Granger**

CENA[1] **CENA**[2] **ENAC**[3]

Keywords: Free Flight, autonomous aircraft, conflict resolution, air traffic control.

Abstract: FACES is an autonomous and coordinated embarked (on board) conflict solver for Free Flight airspace. It solves conflict by computing simple manoeuvres that guarantees conflict free trajectories for the next 5 minutes (min). Coordination is ensured by giving sequential manoeuvres to aircraft with a token allocation strategy. **FACES** can be implemented with the current positioning, broadcasting and flight management technology. Moreover, it is robust to communication or system failure for time up to one or two minutes. **FACES** was tested with a traffic simulator on busy traffic days over France. Airspace over level 320 was considered as Free Flight. 638 out of 641 conflicts were solved without using vertical manoeuvres. The 3 remaining conflicts could easily be solved with vertical manoeuvres. The mean delay was less than 30 seconds by aircraft manoeuvred, with a max delay of 150 seconds. An airborne implementation of this algorithm can be seriously considered.

.

Introduction

We have all experienced at least once a long wait in an overcrowded air terminal. Reading magazines distributed by airlines during these long hours, we often found that they consider air traffic control as one of the major cause for delays. And it is true that the air traffic control system is becoming saturated. But, if delays due to overloaded airports are easy to understand, it is much harder to comprehend delays due to the en route control system. In fact, if we ask a mathematician to analyze the system in cold blood, it can be proved that the collision probability over flight level 320 is very low for aircraft flying direct routes, especially if some elementary precautions are taken regarding face to face or overtaking conflicts. So, en route control could be considered as expensive (en route charges), inefficient (delays induced) and statistically of very little use.
However, if the Free Route and Free Flight concepts are attractive, especially to airlines, we still must consider safety as the first priority, and design new algorithms and systems for these new airspaces(airspace above flight level 320 without control).

The most well known reactive collision avoidance concept is certainly the ACAS/TCAS system. It is a very short term collision avoidance system (less than 60 seconds) that is the last safety filter of an ATC system. Experiments with TCAS to control aircraft on simulated traffic have shown that poor coordination could lead to disastrous

[1] Centre d'Etudes de la Navigation Aérienne, 7 av Ed Belin, 31055 Toulouse, France, e-mail : alliot@recherche.enac.fr
[2] e-mail : durand@recherche.enac.fr
[3] Ecole Nationale de L'Aviation Civile, 7 av Ed Belin, 31055 Toulouse, France, e-mail : granger@recherche.enac.fr

situations, especially when more than 2 aircraft are simultaneously involved [1]. Other simple techniques using repulsive forces [2, 3] have also been investigated but drawbacks remain [1].

This paper presents an algorithm for autonomous embarked conflict resolution with a coordination mechanism. Moreover, it is robust to communication or system failure for times up to one or two minutes. This algorithm could be implemented with current technology (**GPS, FMS, ADS-B**) at low cost.

First section deals with hypothesis we made and the modelling chosen. The second section presents the ordering strategy. In the third section , the A^* algorithm used to optimize a conflict free trajectory for one aircraft is detailed. In fourth section , the basic algorithm is tested with the air traffic simulator CATS [6] on a heavily loaded traffic day in the French airspace over flight level[4] 320. Experimental results led us to introduce slight improvements to the basic algorithm also presented in the fourth section ; then we finally discuss results of simulations on the enhanced algorithm.

1 Modelling

1.1 Hypothesis

The idea is to build an embarked—that is, an on-board—solver able to compute a manoeuvre each time a conflict is detected with another aircraft in a defined detection area around the aircraft. This solver should continuously (every minute[4]) guarantee a 5[4] minute conflict free trajectory to each aircraft. This 5 minute conflict free period guarantees that a transient failure of communications would not have a disastrous effect: the system could still restart later on; resolutions would be less optimal, more vertical manoeuvres could be necessary to solve all conflicts, as anticipation would be shorter, but the risk of collision would remain close to zero.

Manoeuvres suggested have to be simple to understand and to execute. No manoeuvre can be given during the first minute (called the quiescent period) in order to give enough time to the solver to compute a solution and inform the pilot (or directly program the FMS). Moreover, only one manoeuvre can be given to one aircraft during a 5 minute time window, and no manoeuvre can start as long as the previous one is not finished.

The algorithm enforces a global resolution order between conflicting aircraft. The general principle is as follows: the aircraft which is first chooses its trajectory without considering other aircraft. Then, the next aircraft in the priority queue takes this trajectory into account, and computes its own, and so on (see the second section).

Airspace over flight level[4] 320 is considered as a Free Flight airspace. This area is not a so low density area, especially in France. So it is an excellent test zone for a Free Flight solver. All aircraft entering this airspace are supposed to be separated for 5 minutes when entering the Free Flight zone, and are sent back separated for the next 5 minutes when leaving it. All aircraft entering this airspace have to be Free Flight compliant, i.e.:

- they all have synchronous clocks;
- they are able to receive all broadcast information from other aircraft which are within a 90 nmi zone around them (see part three in this section);
- they are all equipped with the FACES solver;
- they are able each minute at the same time to compute, and store their current position, their Free Flight airspace exit point and their predicted trajectory for the next 5 minutes;
- they are able to reliably broadcast the latter information as soon as it has been computed.

This information consists of 20 3D-points, one every 15 seconds (in fact, only 16 are needed, those beginning at t=1 min). Extra information is added to the predicted position that indicates its accuracy (the uncertainty model is detailed in the third part in this section). Of course, the more accurate the information, the more efficient detection and resolution. This prediction has to be *contractual*, i.e. as soon as an aircraft has broadcast the information, it has to keep to this trajectory for the next 5 minutes as long as the solver does not give a manoeuvre. It must be noticed that on exceptional occasions, one aircraft can modify this trajectory, or aircraft not equipped for Free Flight can be accepted in the Free Flight zone. This can also take into account exceptional events such as the failure of one aircraft conflict solver. These aircraft will be given the highest priority number (see second part) and all other aircraft will build their trajectory in order to avoid them. This should be a last resort, as the algorithm might fail if two such aircraft are present at the same time in the same zone.

[4] These parameters can be modified. This first study does not discuss the opportunity of increasing or decreasing these values.

1.2 Manoeuvre modelling

As stated above, time is discretized into 15 seconds[5] time steps. As manoeuvres must remain simple to understand and execute, the turning point modelling is chosen in the horizontal plane (see Figure 1). In this article, no manoeuvre is given in the vertical plane[6] .

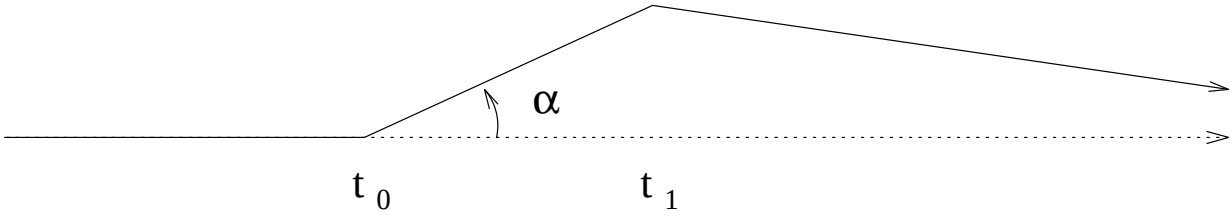

Figure 1: Turning point modelling.

As shown on Figure 1, a manoeuvre is a heading change of 10, 20 or 30 degrees right or left, it starts at time t_0, and ends at time t_1 on the original path. As stated above, t_0 (and t_1) are always larger than 1 minute.

1.3 Uncertainty modelling and 1-to-1 conflict detection

A very simple filter is first applied: only aircraft within a 90 nmi zone are considered as being potential threats. This radius is such that 2 aircraft facing each other at 500 kn cannot be in conflict[7] during the next 5 minutes if they are not in the detection zone of the approaching aircraft.

We then assume that there is an error about the aircraft's future location because of ground speed prediction uncertainties[8] . Uncertainties on climbing and descending rates are even more important[9] . Uncertainties on the future positions of aircraft are all the more important because the prediction is faraway. A 5% of uncertainty on ground speed for an aircraft flying at 480 kn represents 2 nautical miles 5 minutes ahead, and about 6 nautical miles 15 minutes ahead. In the vertical plane, 20% of uncertainty for an aircraft climbing at 2000 feet (ft) per minute represents 2000 ft 5 minutes ahead and 6000 ft 15 minutes ahead.

In the vertical plane, we use a cylindrical modelling (Figure 2). Each aircraft has a maximal altitude and a minimal altitude. To check if two aircraft are in conflict, the minimal altitude of the higher aircraft is compared to the maximal altitude of the lower aircraft.

---

[5] This value is not chosen at random. With 15 s time steps, detection can be made only on these points (and not on the segments between these points) with the guaranty that two aircraft can not cross each other without noticing a serious conflict.

[6] Vertical manoeuvres were put aside on purpose. They are more difficult to execute, and less comfortable for both pilots and passengers. Results of the fourth part show that they should only be used as a last resort, on the very rare occasions where the solver fails.

[7] In this article the separation standards are 6 nmi in the horizontal plane and 1000 ft in the vertical plane.

[8] Uncertainties on ground track will not be considered, as they do not increase with time and will be included in the separation standard.

[9] The error percentages on vertical and horizontal speed are specific to each aircraft. For example, aircraft with very accurate FMS will have very low percentages.

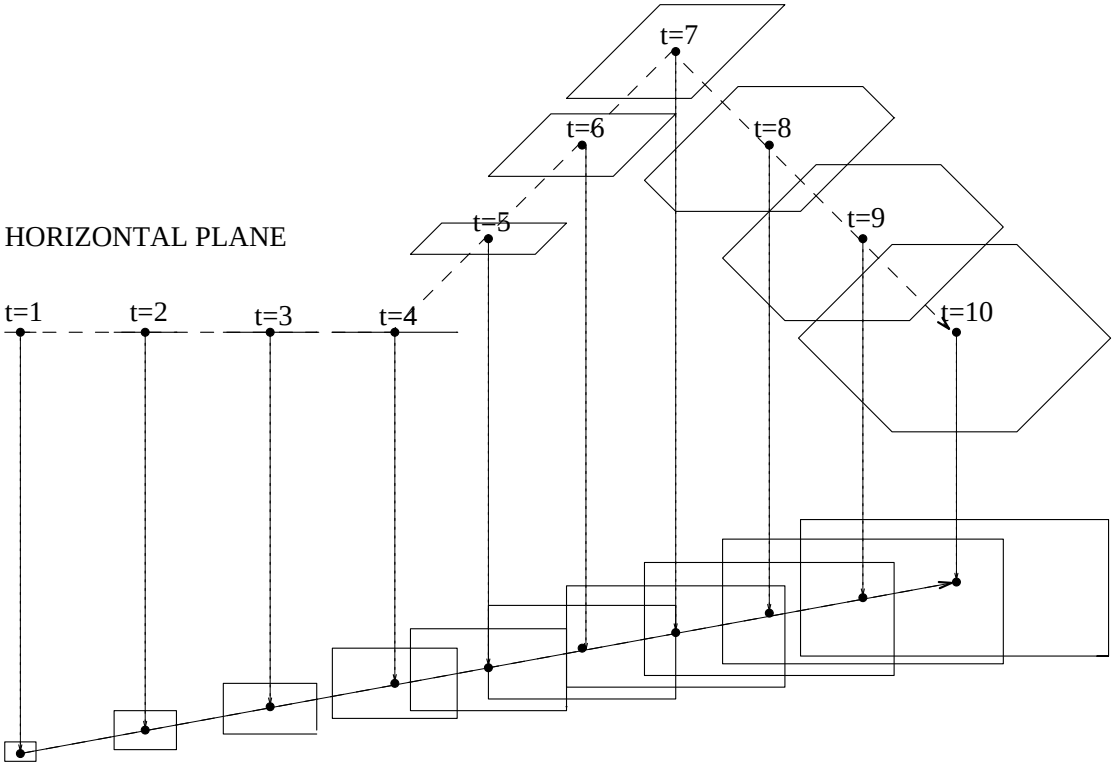

Figure 2: Modelling of speed uncertainties.

In the horizontal plane, an aircraft is represented by a point at the initial time. The point becomes a line segment in the uncertainty direction (the speed direction here, see Figure 2). The first point of the line ``flies'' at the maximum possible speed, and the last point at the minimum possible speed. When changing direction (t=4), the segment becomes a parallelogram that increases in the speed direction. When changing a second time direction (t=7), the parallelogram becomes a hexagon that increases in the new speed direction. To check the separation standard at time t, we compute the distance between the two polygons modelling the aircraft positions and compare it to the separation standard at each time step of the simulation. It must be noticed that, as only one manoeuvre can be given in a 5 minutes time window, and as no manoeuvre can start as long as the previous one is not finished, the convex can only be a line, a parallelogram or an hexagon.

A classical problem in 1 to 1 conflict detection is symmetry. If aircraft A considers it is in conflict with aircraft B, then B must consider A as a conflicting aircraft. In FACES, broadcasting of positions guarantees that two aircraft that can detect each other share exactly the same information regarding their positions. As detection algorithms are identical, 1 to 1 detection will always be symmetrical.

2 Ordering strategy

2.1 The coordination problem

Centralized automatic solvers as described by N. Durand[4] find a global solution to clusters involving many aircraft. Manoeuvres are then given to aircraft simultaneously. An on board solver cannot be based on the same principle: aircraft do not share the same information, as they do not have the same detection zone (limited to 90Nmi). A coordination problem appears and must be solved.

The Free-R [5] project uses extended flight rules to solve this problem. The TCAS system uses the transponder code to decide which aircraft has to manoeuvre; giving resolution priorities to aircraft is a way often adopted for solving the coordination problem.

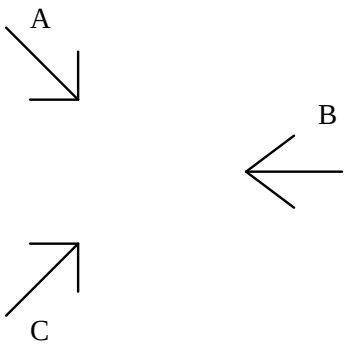

Figure 3: 3 aircraft conflict.

A resolution priority order[10] has to be total if we want each aircraft to solve all conflicts when there is more than 2 aircraft. For example, the Visual Flight Rule that gives priority to the aircraft coming from the right does not define a global order if there are more than 2 aircraft simultaneously in conflict. Figure 3 gives an example of conflict involving 3 aircraft for which this priority resolution rule does not define an order because transitivity is not ensured.

Moreover, as aircraft do not share the same information, defining which aircraft will be the first to choose its trajectory is not obvious, even if we have a total priority order. Let's take an example.

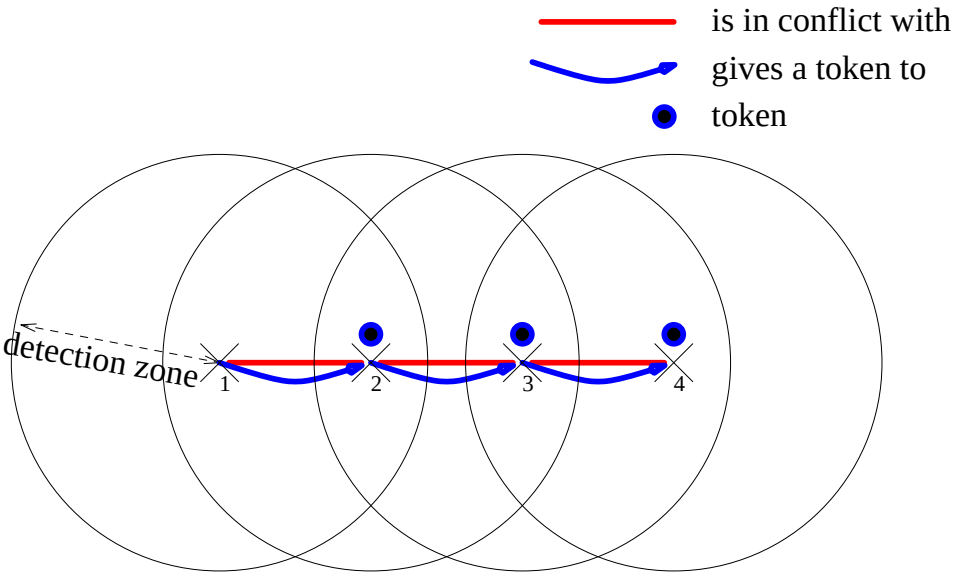

Figure 4: 4 aircraft cluster.

On Figure 4, four aircraft are in conflict. Aircraft 1 is in conflict with aircraft 2 which is in conflict with aircraft 3 which is in conflict with aircraft 4. However, aircraft 1 can only detect aircraft 2 which detects aircraft 1 and 3. Aircraft 3 detects aircraft 2 and 4 which detects aircraft 3. Let's use a simple priority order between these 4 aircraft as for example (1>2>3>4). In conflict (1,2), aircraft 2 will have to manoeuvre. In conflict (2,3), aircraft 3 will have to manoeuvre and in conflict (3,4), aircraft 4 will have to manoeuvre. However, aircraft 4 does not detect aircraft 2. Consequently, it can not know that aircraft 3 has to manoeuvre before it can manoeuvre itself. Furthermore, aircraft 3 does not detect aircraft 1 and can not know that aircraft 2 has to manoeuvre before it can manoeuvre itself. New trajectories computed independently can then be inconsistent at the end of the resolution step. So, it is not possible to directly derive the resolution order from a simple priority order based only on transponder code for example.

2.2 Building a global resolution order

2.2.1 A token allocation strategy

We have to define a global resolution order such so that each aircraft can know when it can start to build a conflict free trajectory and which aircraft it has to avoid.

We first suppose that a total priority order relation exists on the aircraft population. We can use the simple order based on transponder numbers discussed above (but more elaborate orders will be discussed later). At each

[10] An order relation must be anti-symmetrical and transitive. An order relation is total if every pair of individuals can be compared.

resolution step, we build a global resolution order from this priority order with the following strategy:

1. First, every aircraft sends its predicted trajectory to its neighbours. Each aircraft is then able to know whether it is conflicting with another aircraft or not for the next five minutes.

2. Each aircraft receives a token from every conflicting aircraft that has a higher priority in its detection zone. Aircraft that are not in conflict never receive any token. In the example of Figure 4, aircraft 2 receives a token from aircraft 1, aircraft 3 receives a token from aircraft 2 and aircraft 4 receives a token from aircraft 3.

3. Then, each conflicting aircraft with no token solves conflicts with every aircraft in its detection zone that has no token. It does not take into account aircraft that have one or more tokens.

4. When this trajectory has been computed, the aircraft broadcasts its new trajectory; all aircraft which have received a token from this aircraft take this new trajectory into account, and cancel the token received from this aircraft.

5. Steps 2 and 3 are repeated until no token remains.

On Figure 4, aircraft 1 does not change its trajectory as it has no zero token aircraft in its detection zone. It broadcasts its new (unmodified) trajectory. So aircraft 2 takes this trajectory into account and cancels the token received from aircraft 1. Then aircraft 2 solves its conflict with aircraft 1 as aircraft 1 is in its detection zone and has no token. Aircraft 3 takes this trajectory into account, cancels token from aircraft 2 and solves the conflict with aircraft 2. Aircraft 4 then applies the same procedure.

2.2.2 Detailed example

The following example has been observed in the simulations that will be presented later. Eight aircraft belong to the same cluster (Figure 5). The detection area of each aircraft is given on the figure.
Aircraft A7 detects every other aircraft. A8 detects A5, A6 and A7. A4 detects A1, A2 and A7. A6 detects A2, A3, A7 and A8. A2 detects A1, A3, A4, A6 and A7. A3 detects A2, A6 and A7. A1 detects A2, A4 and A7. A5 detects A7 and A8. Conflicting aircraft are A1-A2, A1-A4, A2-A3, A2-A7, A3-A6, A5-A7 and A5-A8. Aircraft that has the highest priority is A1 and the lowest priority order is A8 (A1 > A2 > A3 > A4 > A6 >A7 > A8).

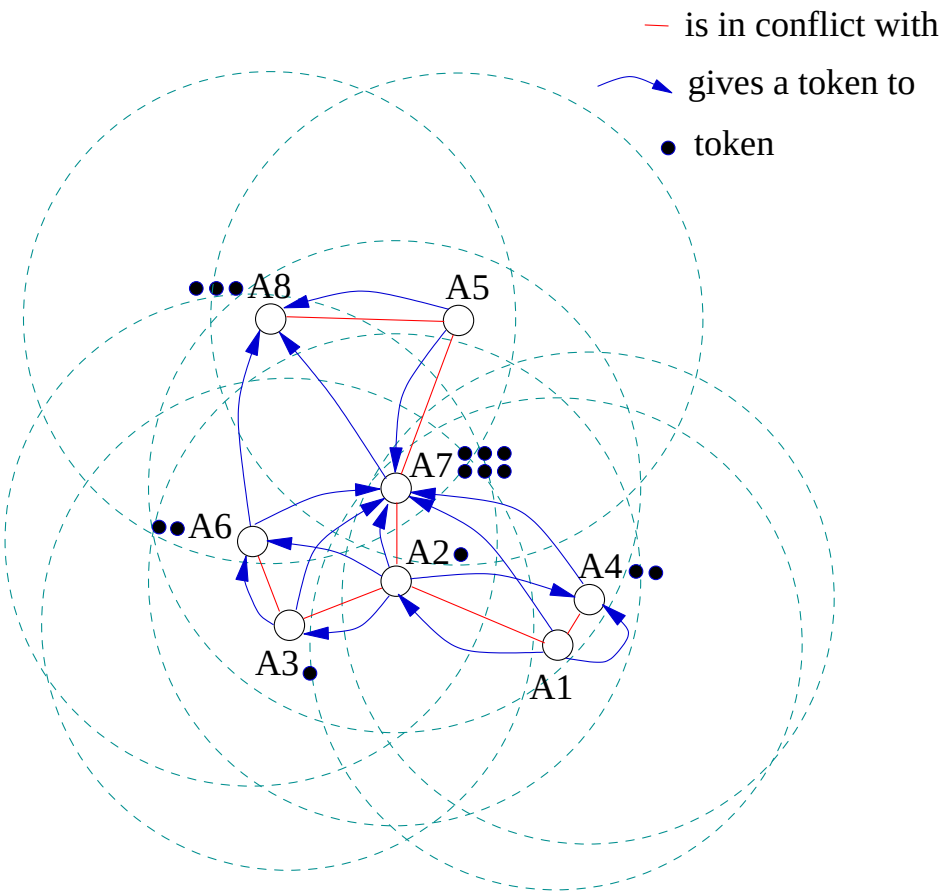

Figure 5: Cluster of aircraft in conflict

Tokens are allocated as follows:

- A1 gives a token to: A2, A4, A7;
- A2 gives a token to: A3, A4, A6, A7;
- A3 gives a token to: A6, A7;
- A4 gives a token to: A7;
- A5 gives a token to: A7, A8;
- A6 gives a token to: A7, A8
- A7 gives a token to: A8;
- A8 gives no token.

Step	1	2	3	4	5	6
A7	6	4	3	1	0	
A8	4	3	3	2	1	0
A4	2	1	0			
A6	2	2	1	0		
A2	1	0				
A3	1	1	0			
A1	0					
A5	0					

Table 1: Token allocation at the different steps of resolution

Table 1 gives the token allocation at the different steps of the resolution.
During step 1, A1 and A5 (0 tokens) choose their trajectories without considering other aircraft in their detection area (those that have at least 1 token). Then, they broadcast their unmodified trajectories and all aircraft that have received a token from them cancel it. A2, A4, and A7 cancel the tokens sent by A1. A7 and A8 cancel the tokens sent by A5. During step 2, A2 (0 token) has no token and modifies its trajectory to solve conflict with A1 (0 token). A3, A4, A6, and A7 cancel the tokens sent by A2. During step 3, A3 (0 token) modifies its trajectory to solve conflict with A2 (0 token), A4 (0 token) modifies its trajectory to solve conflict with A1 (0 token); the new trajectory must not interfere with A2 (0 token). A6 cancels one token sent by A3 and A7 cancels two tokens sent by A3 and A4. During step 4, A6 (0 token) modifies its trajectory to solve conflict with A3 (0 token), the new trajectory must not interfere with A2 (0 token). A7 and A8 cancel one token sent by A6. During step 5, A7 (0 token) modifies its trajectory to solve conflict with A2 (0 token) and A5 (0 token); the new trajectory must not interfere with A1 (0 token), A3 (0 token), A4 (0 token) and A6 (0 token). A8 cancels the token sent by A7. During step 6, A8 (0 token) modifies its trajectory to solve conflict with 5 (0 token); the new trajectory must no interfere with A6 (0 token) and A7 (0 token).

2.3 Provability

The allocation-resolution method described above cannot lead to situations where all aircraft would have at least one token or situations where two aircraft detecting each other without any token would have to solve simultaneously. This is guaranteed by the use of a total priority order on aircraft. At each step, an aircraft with no token cannot have any other conflicting aircraft (that has not already solved) with no token in its detection area. In such a case, one of these two aircraft would have given a token to the other. At each step, among the conflicting aircraft that have not already solved, there is one that has the highest priority. This aircraft cannot have any token. It can solve and get back its tokens. The algorithm can be mathematically proved.

2.4 Communications

The token allocation method gives a mental picture of how aircraft coordinate themselves. Practically, as stated earlier, no bilateral communication is necessary. If aircraft clocks are synchronous, with the information broadcasted, every aircraft can know how many tokens it has and know its neighbour's trajectories. When an aircraft ends a resolution, it broadcasts its new trajectory. Aircraft that have (virtually) received a token can then cancel it.

It is interesting to give a rough estimate of the capacity needed for the communication channel. As stated previously, 16 ``points" are broadcast. Each point can be as complex as a regular hexagon (see uncertainty modelling), with a lower and a higher altitude. A regular hexagon can be defined by only 4 (x,y) points; if we consider a **GPS** resolution of 100 meters, given that the earth circumference is 40000 km, x or y value can be coded with 20 bits. For the sake of simplicity, we will also assume that z values are coded on 20 bits. So, a predicted trajectory will consist of at most 16 _ (4 _ 2 +2) _ 20 = 3200 bits. Some other information has to be broadcast, such as the transponder code, the Free Flight zone exit point, etc. This gives a maximum of 3500 bits.

To implement such a system, the ADS-B messages would have to be extended to include trajectory broadcast. But both STDMA and mode-S would provide enough bandwidth. The 1 Mbits/s capacity of mode-S, even divided by 10, would enable enough re-emissions of messages to guarantee a high level of reliability. Regarding STDMA, it solves the garbling problem and the 4000 slots of 150 bits in 1 minute would give 4 slots of 3000 bits in 1 minute to 50 aircraft in the same zone, twice the capacity needed. Of course, a fine modelling of these systems is required to correctly estimate reliability and availability.

3 The A^* algorithm

As soon as the resolution order is chosen, the problem is to solve a 1 to n conflict problem: we have to find the minimum length trajectory for an aircraft avoiding n already fixed aircraft trajectories, that can be considered as obstacles. This is a classical robotics problem, therefore a classical A^* algorithm (see [9]) is used. The A^* algorithm finds the shortest path in a tree, given an initial state and a set of final states. It uses a ``best first" strategy to develop each node. The best node is the one that provides the lowest expected cost.

To use an A^* algorithm we need:

- u_0: the initial state;
- T: the set of terminal states;
- $p_1,p_2,..,p_n$, the set of rules used to build a leaf from a node; ($v \leftarrow^{p}_{i} u$) means that v is built from u using the rule p_i;
- $k(u,v)$: if u and v are two nodes, k is the cost of the arc (u,v). The total cost of a state is the sum of the costs of the arcs used to connect the initial state to the current state. In this application, the cost of an arc is the corresponding trajectory length;
- $h(u)$: if u is a node, h is a heuristic that estimates the minimal necessary cost to connect the current state to the final state.

The efficiency of the A^* algorithm depends on h. For a state u, $h(u)$ tries to approach:

$$h^*(u) = Min_{(v_1,v_2,..,v_n)} [\, k(u,v_1)+k(v_1,v_2)+... +k(v_n,u_t)]$$

with u_t a terminal state.

A heuristic is perfect if:

$$\forall\ u, v, \quad h(u)=h(v) \Leftrightarrow h^*(u) = h^*(v)$$

A heuristic is underestimating if:

$$\forall\ u, \quad h(u) \leq h^*(u)$$

The performance of the algorithm strongly depends on the quality of the heuristic chosen. With a perfect heuristic, the first path developed is the optimal path; a underestimating heuristic guarantees that the algorithm always converges to the best solution.

In the present application, the initial state is the state of the solving aircraft at t=1 minute. The terminal states are the possible states of the solving aircraft after 5 minutes of flight or when they have reached their destination.

Each branch of the tree represents a possible trajectory of the solving aircraft. Fortunately, the heuristic function is used to only develop a small part of the tree.

The cost of a path is the trajectory length described by this path. Before starting a manoeuvre, an aircraft is in S_0 state. At each time step, each S_0 state generates 6 S_1 states corresponding to the 6 possible deviations of the trajectory (10, 20, 30 degrees right or left), and 1 S_0 state (the aircraft is not manoeuvred). At each time step, each S_1 state generates one S_1 state (the manoeuvre is extended) and one S_2 state (the aircraft is sent back to its Free Flight zone exit point). Every state generates a terminal S_3 state after 5 minutes or if the aircraft has reached its destination.

The cost function $k(u,v)$ measures the distance between the position of the aircraft at node u (time step t_{tu}, state S_{su}) and the position of the aircraft at node v (time step t_{tv}, state S_{sv}). If a conflict occurs between node u and node v, the value $k(u,v)$ is widely increased so that the corresponding branch is no longer developed.

The heuristic function $h(u)$ is here the direct distance between node u and the Free Flight exit point (destination) of the solving aircraft. This heuristic is clearly a underestimating one, which guarantees that the optimal solution will always be found.

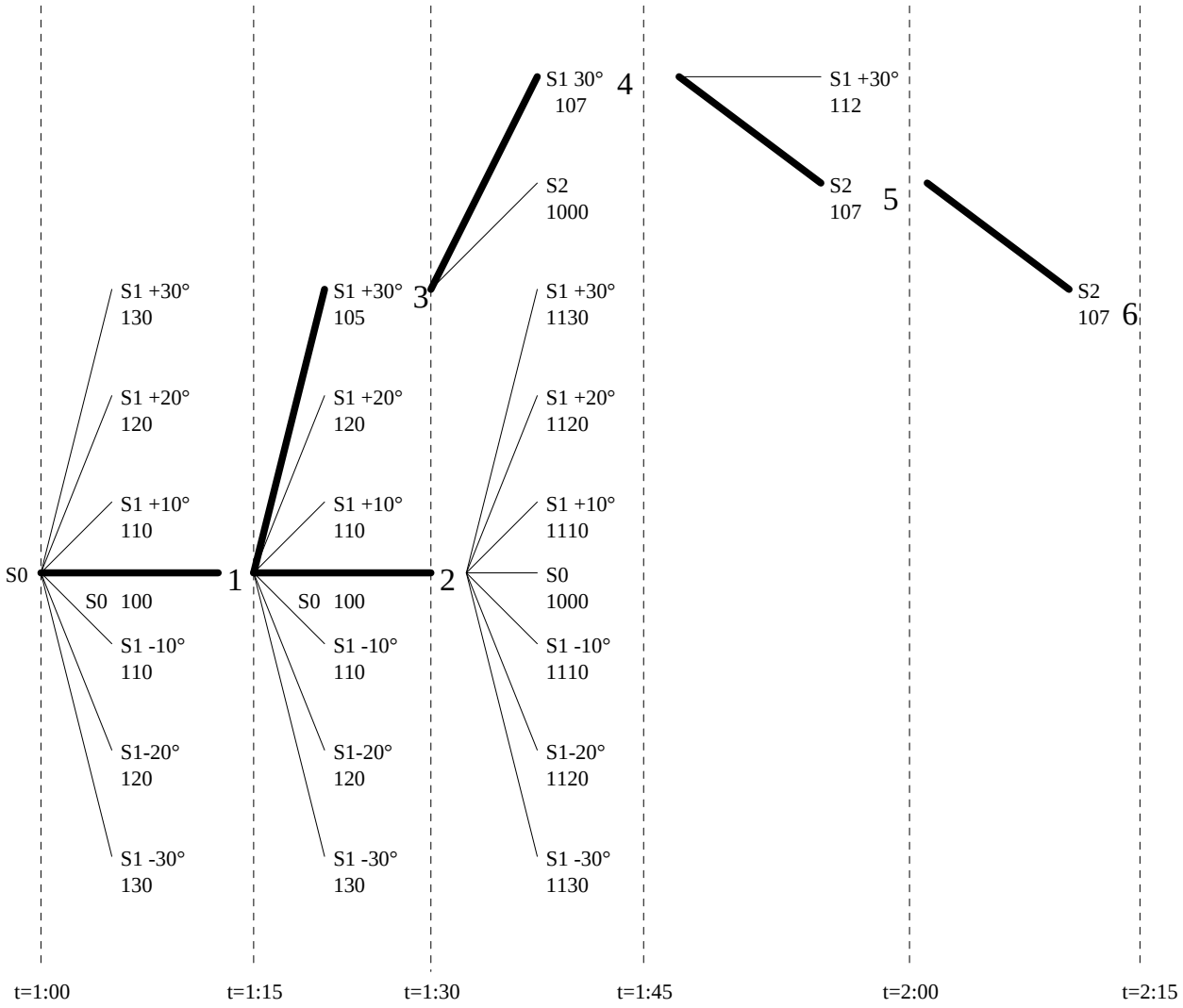

Figure 6: Example of manoeuvre optimization.

Figure 6 details an example of a search in a tree. In this example, the solving aircraft has no manoeuvre at time t=1 minute (t=1 minute corresponds to the beginning of the optimization), it is in S_0 state. The first node chosen is the node that has the lowest estimated cost (h+k). In the example, the lowest estimated cost (100) is found if no manoeuvre is chosen at time 1 minute (step 1). At time 1:15 the lowest estimated cost is still 100 if no manoeuvre is chosen (step 2). At time step 1:30 the lowest estimated cost is 1000 because of a conflict occurring between 1:30 and 1:45 whatever the manoeuvre chosen is. Consequently the branch that has the lowest estimated cost is then developed. Here, at time 1:15 a manoeuvre of 30 degrees left gives an estimated cost of 105 (step 3). The solving aircraft is moved to S_1 state. At the next step, the lowest estimated cost is found if the manoeuvre is extended (107). The alternative choice (S_2 state: turn back to the destination) generates a conflict (step 4). At time 1:45, the lowest estimated cost is found if the aircraft turns back to the destination (S_2 state, step 5). At t=2:00, only one state (S_2) can be generated and its estimated cost is still the lowest (step 6).

Generally, many different paths are developed and the depth of the tree is 16 (4 minutes). In this applications, the solution is given in less than 5 seconds on a Pentium II 300, even for the biggest 1-to-35 conflict.

4 Experimental results and improvements

4.1 The CATS simulator

The algorithm was tested on the **CATS** [6] simulator. The core of the CATS system is an en route traffic simulation engine. It is based on a discrete, fixed time slice execution model: the position and speed 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 burn as a function of altitude, aircraft type and flight segment (cruise, climb or descent.) Two main datasets for aircraft flight performance are used:

- CAUTRA / ENAC aircraft performance tables, extracted from the French flight data processing system;
- Base of aircraft data (BADA) performance summary tables derived from the total energy model of EUROCONTROL. 69 different aircraft types are described. Synonym aircraft are used to model the rest of the fleet. The Airbus A320 (EA32) is used as default aircraft.

In the further applications, aircraft use direct routes to their destination. The separation standard used is 6 nautical miles in the horizontal plane and 1000 ft vertically[11] . Conflicts were not solved under flight level 320, and a delay t_e was added when necessary for aircraft entering the Free Flight zone in order to separate them on entry points. Uncertainties on speed (either vertical or horizontal) were set to minimal values.

4.2 Results using an arbitrary order

Results presented in this part are obtained with the 6381 flight plans of the 21st of June 1996 with no regulation. The Free Flight zone defined is the airspace above flight level 320. The allocation-resolution strategy described earlier is repeated every minute and the trajectory prediction is done on the next 5 minutes.

2763 aircraft enter the Free Flight zone. 641 conflicts are detected in this zone during the day.

The algorithm requires the definition of a total order among aircraft. The simplest order that can be chosen is an arbitrary order based on the **CAUTRA** number of each aircraft. This kind of order is already used in the **TCAS** system.

The A^* algorithm is called 2781 times, which represents a mean of 4 times per conflict. At the end of the simulation, 104 conflicts remain. The simulations lasts only a few minutes.

The failing cases were investigated and most of the time the following situation appeared: an aircraft that has a low priority order starts a manoeuvre. A few minutes later, this aircraft is involved in a new conflict while its manoeuvre is not finished. As it still has a low priority order (its **CAUTRA** number has not changed), it has to solve the conflict. It cannot find a conflict free manoeuvre because the started manoeuvre can not be called into question.

This situation led to the following conclusion: an aircraft that has already started a manoeuvre should have a higher priority order than an aircraft that has not started any manoeuvre. This rule was added to the previous order definition and the simulation was executed again.

4.3 Manoeuvre free priority order

The new total priority order defined in this section is the following: an aircraft that is manoeuvre free has a lower priority order than an aircraft that has already started a manoeuvre. The **CAUTRA** number is used to compare two manoeuvre free aircraft or two manoeuvred aircraft.

With this new priority order, the A^* algorithm is called 2654 times. At the end of the simulation, 29 conflicts remain.

Most of the remaining conflicts are due to the horizon effect[12] already observed by N. Durand in [7]. The criteria optimized by the A^* is the trajectory length between the initial state and the terminal state at the end of the time window (5 minutes). For converging aircraft, the criteria is very often minimized by postponing the conflict out of the time window instead of solving it. Therefore, the solving aircraft is given a heading that tries to move it away

[11] The 6 Nmi value is certainly quite high for an embarked solver. We do think that it could seriously be reduced, regarding GPS and FMS precision. This would even improve airspace capacity and resolution efficiency.

[12] This effect has been observed and discussed for years now in other tree search algorithms: game algorithms.

from the other aircraft. Figure 7 gives an example of such a conflict: aircraft 3467 is first turned left to move away from aircraft 3660. 1 minute later, aircraft 3660 becomes the solving aircraft and turns right to move away from aircraft 3467. 1 minute later, as both aircraft are not manoeuvre free, the conflict can not be solved.

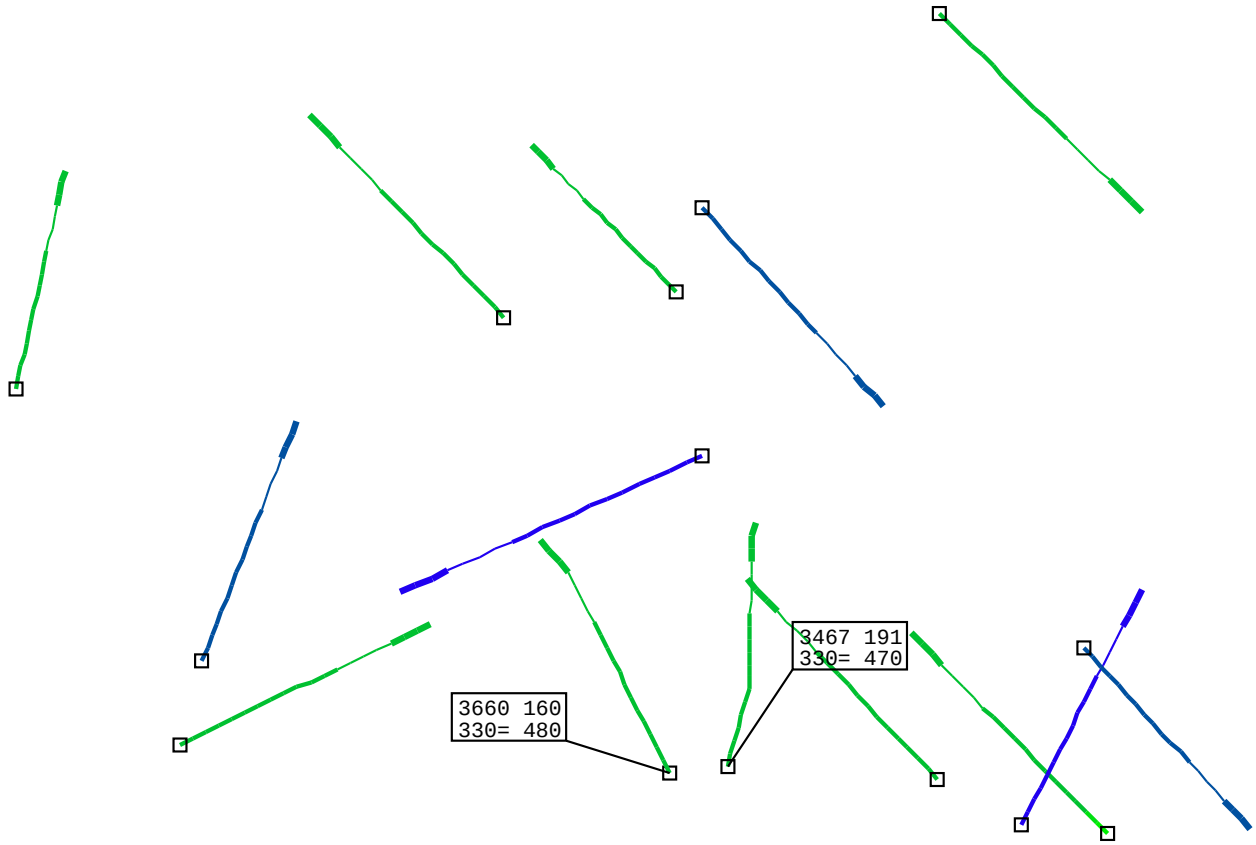

Figure 7: Second step: aircraft 3660 turns right

To solve this problem the cost criteria used by the A^* algorithm must be changed to take into account the efficiency of a manoeuvre. Therefore, when two aircraft must cross, the manoeuvre that enforces crossing must have a lower cost than the manoeuvre that postpones the crossing. This new criteria is included in the A^* algorithm.

4.4 Manoeuvre efficiency criteria

In this section, the order previously defined is used and the cost function is modified to take into account the efficiency of the manoeuvre as defined above. The simulation is executed once again. Three conflicts remain unsolved, but they can easily be solved by a very simple vertical manoeuvre.

4.5 Delays

897 manoeuvres are given during the day to 367 aircraft (2.44 manoeuvres per aircraft). 13.28% of the aircraft flying in the Free Flight zone are manoeuvred.

73 aircraft are delayed when entering the Free Flight zone, because they are not conflict-free at that moment (that is 2.64% of the traffic in the Free Flight zone).

Delays are given in table 2.

	Mean delay Per aircraft	Mean delay per delayed aircraft	Max delay
Delayed	2*s*	1*min* 14*s*	3*min*
Manoeuvred	3.6*s*	27.4*s*	40*s*

Table 2: Delays

The maximum manoeuvre lasts 2:30 minutes.

Table 3 gives the number of steps aircraft have to wait because they have been given tokens. In the most complex conflict, one aircraft has to wait for 7 resolutions before it can choose its trajectory.

Number of Steps	Waiting Aircraft	Percentage of Total
1	1516	76,6%
2	253	12,78%
3	162	8,19%
4	27	1,36%
5	17	0,86%
6	3	0,16%
7	1	0,05%

Table 3: Number of waiting aircraft.

So, regarding delays, the performance of the algorithm is very good.

4.6 Unsolved conflicts and priority order

There are 3 aircraft in each remaining unsolved conflict. These conflicts appear because the order between aircraft is not well chosen. F. Médioni [*8*] in his PhD thesis showed that a very good solution or no solution at all could be found for a simple conflict involving only 3 aircraft, depending on the order chosen. This situation is described in Figures 8, 9 and 10. Moreover, it looks extremely difficult to devise an algorithm that would find the best possible order without seriously increasing the complexity of the global algorithm and the necessary capacity of the communication medium. Embarked conflict solvers which have only a partial information on the global situation will almost certainly remain suboptimal, while centralized conflict solvers are able to find the global optimal solutions.

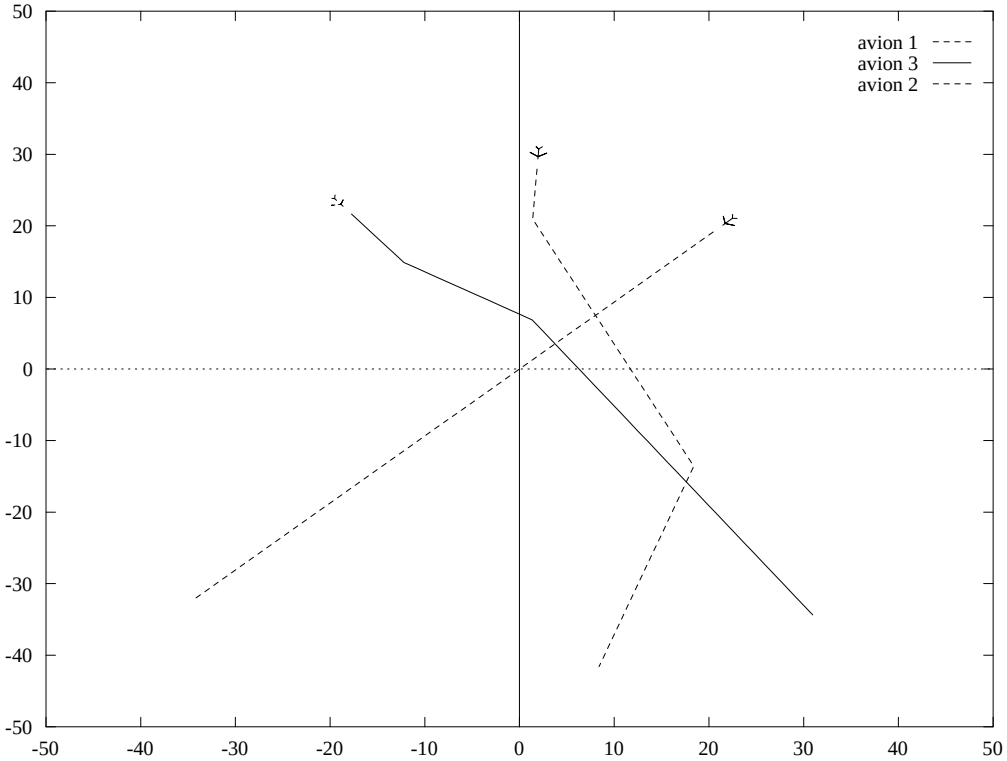

Figure 8: order (1,2,3), optimal solution

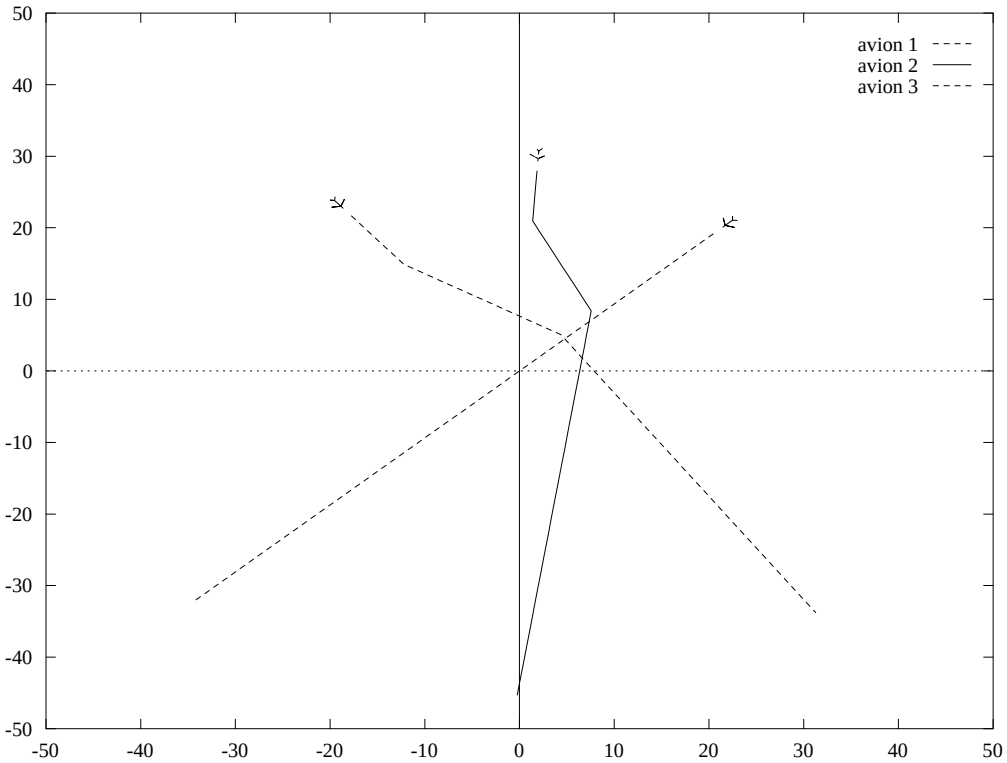

Figure 9: order (1,3,2), suboptimal solution

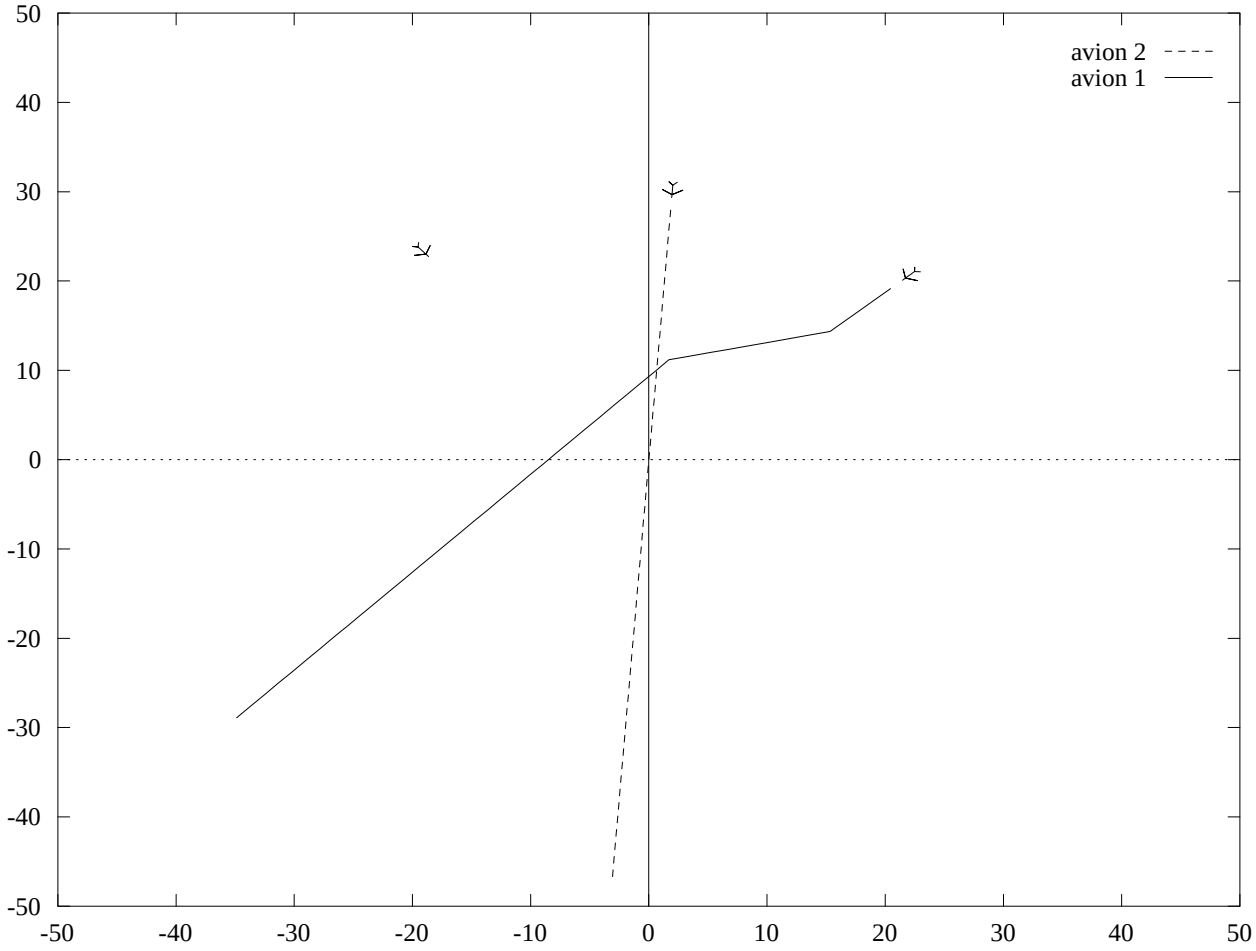

Figure 10: order (2,1,3), no solution.

However, this may not be a too serious concern in the upper airspace: the simulation above shows that this algorithm is almost always able to solve conflicts, even with situations as complex as the one presented on on Figure 11 where 35 aircraft are involved, while delays remain small.

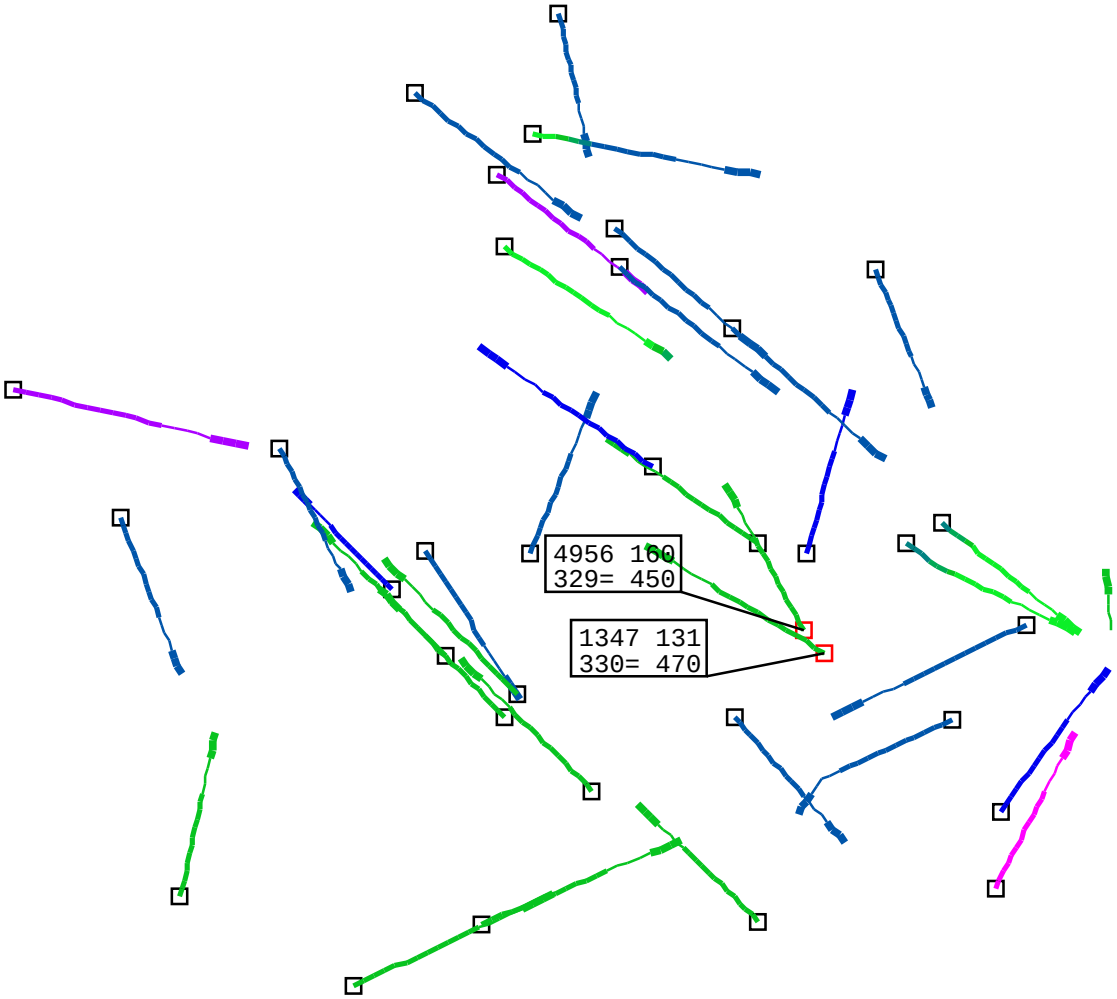

Figure 11: 35 aircraft in the detection zone.

5 Conclusion

We have demonstrated in this paper that an efficient on board algorithm for Free Flight conflict resolution can be designed and implemented. This algorithm has the following advantages:

- Compared to a centralized automated system, the development of such a system could be relatively low cost. Most assumptions are quite weak: synchronous clocks are already available with **GPS**, **FMS** are now elaborate enough to provide the information needed for trajectory prediction in the next 5 minutes, **ADS-B**, or even better **STDMA**, provides sufficient capacity for communications; the 1 to *n* resolution algorithm is simple to implement and has already been widely used for similar problems in robotics; computing power needed fits in a standard PC computer.
- Compared to rule based systems, the algorithm is mathematically provable, and the simulation above shows that it would be efficient in upper airspace, even when the density is quite high, and even with quite a large separation standard: without vertical manoeuvres, only 3 conflicts out of 641 remain. 13% of the aircraft are manoeuvred and the mean delay is 27 seconds by aircraft effectively manoeuvred, with a maximum delay of 150 seconds.
- Compared to purely reactive systems [2], which usually require constant changes in headings, the manoeuvre model is classical and easy to implement. Further, and this is the main point to stress, as trajectories are guaranteed conflict free for at least 5 minutes, a transient failure of communications would

not have a disastrous effect: the system could still restart later on; resolutions would be less optimal, more vertical manoeuvres could be necessary to solve all conflicts, as anticipation would be shorter, but the risk of collision would remain insignificant.

- The system could be progressively put into service by first defining Free Flight airspace over oceanic areas, and gradually extending them. This would help in solving the classical transition problem from the current system to a partially automated one.

The main issue that has to be refined is the value of the quiescent time window (set to one minute in our simulation). During this period, each aircraft has to build its predicted trajectory and broadcast it. Then the whole loop of the algorithm has to be executed, with resolutions, new trajectories broadcast, etc. It is still unclear if one minute will be enough. Simulations show that one minute is a correct value when communications are instantaneous. A fine modelling[13] of the communication medium is needed to confirm or deny this value. However, simulations could be conducted with larger values to test the behaviour of the algorithm.

Simulations have also to be conducted using larger uncertainty parameters. As trajectory prediction is done for the next 5 minutes only, results should not be very different. We will also test the solver with higher traffic to find out its limits regarding density. Vertical manoeuvres will be added as last resort to correctly complete the resolution step.

We are conscious that the whole system depends on the reliability and availability of transmissions. Requirements on the bandwidth are low enough to enable multiple emissions of messages. But error correlations would have to be considered. Specific data and results on these issues do not appear to be presently available. However, we believe that an airborne implementation of this algorithm can be seriously considered.

[13] This modelling would also be very useful to find out the exact availability, capacity and error rate of the channel.

References

[1] Jean-François Bosc. *Techniques d'évitement réactif et simulation du trafic aérien*. PhD thesis, ENAC, 1997.

[2] Karim Zeghal. *A review of different approaches based on force fields for airborne conflict resolution* AIAA Guidance, Navigation and Control Conference, Boston, august 1998.

[3] Karim Zeghal. *A comparison of different approaches based on force fields for coordination among multiple mobiles* IEEE International Conference on Intelligent Robotic System (IROS), Victoria (BC), Canada, October 1998.

[4] Nicolas Durand. *Optimisation de Trajectoires pour la Résolution de Conflits en Route*. PhD thesis, ENSEEIHT, 1996.

[5] V. N. Duong, E. Hoffman, J. P. Nicolaon. *Autonomous Aircraft* 1st U.S.A/EUROPE ATM R & D Seminar, SACLAY 1997

[6] J.M. Alliot, N. Durand, J.F. Bosc, L. Maugis. *CATS: a complete air traffic simulator*. 16th AIAA/IEEE Digital Avionics Systems Conference, IRVINE 1997

[7] N. Durand, J.M. Alliot. *Optimal Resolution of En Route Conflicts*. 1st U.S.A/EUROPE ATM R & D Seminar, SACLAY 1997

[8] Frédéric Médioni. *Méthode d'optimisation pour l'évitement aérien : systèmes centralisés, systèmes embarqués*. PhD thesis, Ecole Polytechnique, 1998.

[9] Judea Pearl *Heuristics* Addison-Wesley, 1984

Acronyms

ACAS : Airborne Collision Avoidance System
ADS-B : Automatic Dependent Surveillance - Broadcast
ATC : Air Traffic Control
BADA : Base of Aircraft DAta
CAUTRA : Coordinateur Automatique du TRafic Aérien
CATS : Complete Air Traffic Simulator
CENA : Centre d'Etude de la Navigation Aérienne
ENAC : Ecole Nationale de l'Aviation Civile
FACES : Free-flight Autonomous Coordinated Embarked Solver
FL : Flight Level
FMS : Flight Managment System
GPS : Global Positionning System
STDMA : Self organising Time Division Multiple Access
TCAS : Traffic alert and Collision Avoidance System

Symbols

nmi : nautical miles
ft : feet
min : minutes
kn : knot
s : second
km : kilometer