Formal Verification of Conflict Detection Algorithms

Ricky Butler[1], Víctor Carreño[1], Gilles Dowek[2], and César Muñoz[3]

[1] Assessment Technology Branch, Mail Stop 130, NASA Langley Research Center
Hampton, VA 23681-2199
{r.w.butler,v.a.carreno}@larc.nasa.gov
[2] INRIA, Domaine de Voluceau - Rocquencourt - B.P. 105,
78153 Le Chesnay Cedex, France
gilles.dowek@inria.fr
[3] ICASE, Mail Stop 132C, 3 West Reid Street, NASA Langley Research Center
Hampton VA 23681-2199
munoz@icase.edu

Abstract. Safety assessment of new air traffic management systems is a main issue for civil aviation authorities. Standard techniques such as testing and simulation have serious limitations in new systems that are significantly more autonomous than the older ones. In this paper, we present an innovative approach for establishing the correctness of conflict detection systems. Fundamental to our approach is the concept of *trajectory*, which is described by a continuous path in the x-y plane constrained by physical laws and operational requirements. From the model of trajectories, we extract, and formally prove, high level properties that can serve as a framework to analyze conflict scenarios. We use the AILS (Airborne Information for Lateral Spacing) alerting algorithm as a case study of our approach.

1 Introduction

Due to rapid growth in air travel, air traffic congestion has become an international problem and it is expected to worsen in the next two decades. Many concepts have been proposed to alleviate this problem. In many of these concepts – such as free-flight [9]–, the responsibility for aircraft separation is partially or completely moved from a centralized air traffic controller to a distributed aircraft system. The consensus in the avionics community is that the distribution will increase the efficiency of the air space system and terminal areas. However, a major concern of engineers, scientists, and civil aviation authorities is that the implementation of new approaches may compromise the overall system safety.

In the avionics community, testing and simulation are the standard methods for assuring the safety of digital systems. For instance the *Airborne Information for Lateral Spacing (AILS)* algorithm [10] - a conflict alerting algorithm used for parallel landing without control of the traffic controller authorities - has been extensively simulated and tested. So far, no major flaws have been detected.

However, neither testing nor simulation can give a definitive answer to questions such as: *"Does there exist a trajectory leading to a conflict without an alarm being issued?"* or *"What is the safety time between a conflict and a prior conflict detection?"* Given the critical nature of the problem, we believe that such analysis should be mechanically checked via a theorem proving system, such as PVS, or other automated proving techniques, e.g., model-checking.

Programs that are used for the monitoring and control of physical devices are constantly interacting with the physical world. The specification of such programs cannot usually be expressed as a mere input/output relation, but also involves concepts related to the physical environment where they are deployed. Hence, proving the correctness of such a system requires reasoning not only about the system itself, but also about its physical environment. For instance, to prove that a conflict detection algorithm does not allow conflicts without first issuing an alarm, we need to reason not only about the properties of the algorithm but also about the properties of the trajectories of the aircraft.

An air traffic control system, such as a conflict detection algorithm, is a hybrid system. It consists of simultaneous discrete and continuous behaviors. The discrete behavior is inherent to the algorithmic implementation on an embedded digital computer. Whereas, the continuous behavior arises from the kinematics of the aircraft. It is, of course, possible to discretize the continuous trajectory of an aircraft, approximating it by discrete segments of trajectories. Discretization of continuous trajectories allows the definition of trajectories by a transition relation. This way, the property we want to prove can be rephrased as a finite state automaton reachability property and it can be established by model checking and/or formal theorem proving. More or less complex extensions to this approach have been often used in the literature to model air traffic control problems (for a list of works, see [11]). These techniques have been shown to be effective for modeling systems where control logic modes trigger continuous and dynamic changes of the state. For instance, the TCAS alerting system for preventing midair collision was modeled in [6] using a hybrid automata approach.

From our experience, the formal verification task can considerably profit from standard mathematical analysis when a continuous model, rather than a discrete one, is used. Indeed, we have used semi-discrete trajectories in a previous effort to verify the AILS algorithm [2]. In that effort, it was not possible to take advantage of many useful properties of elementary calculus to prove the desired properties. For example, minimums and maximums of a differentiable function can easily be obtained by computing the derivative of the function, making it equal to zero, and solving for time. To find minimums and maximums of a function in a semi-discrete model requires an extraordinary effort. Moreover, introducing such a discretization of time and space may complicate the reasoning (it is more complicated, for instance, to reason about floats than about real numbers, because properties such as associativity of addition do not hold for floats).

In this paper, we propose an innovative approach, based on the formalization of continuous mathematics within a higher-order logic framework, for establishing the correctness of conflict detection systems. In this approach, a trajectory is given by differentiable functions in $\mathbb{R} \to \mathbb{R}$, which map *time* into each one of the components of a state aircraft, e.g., *heading*, *position*, *bank angle*, and *ground speed*. Using standard calculus, geometry, and kinematics, we formally asses safety properties of conflict detection algorithms.

In [5], Kuchar and Yang characterize three kinds of trajectory models: *nominal*, *worst-case*, and *probabilistic*. In the nominal approach, the future aircraft state, i.e., position, speed, heading, bank angle, is projected from the current state according to physics laws. In the worst-case approach, the future state is projected by following a policy of extreme values for specific state variables. In a probabilistic model, uncertainties such as weather conditions or extrapolation errors are taken into account to calculate the most probable aircraft trajectories. In our higher-order logic formalization, we can quantify over all (nominal) trajectories and thus worst-case trajectories are just particular cases of nominal trajectories.

Using our approach, we have been able to prove that the AILS algorithm is *correct*, i.e., (1) there is no possible conflict between two aircraft without a prior alarm and (2) the time between an alarm and a potential conflict is at least 10 seconds. We have also proven that the AILS alerting algorithm is not *certain*, i.e., an alarm can be issued when no potential conflict exists. Although AILS will be our running example, we believe that the continuous model of aircraft trajectories we build in this paper is rather general and may be used to prove properties of other algorithms and concepts.

The remainder of this paper is organized as follows. In Section 2, we present a model of continuous trajectories, which is the core of the framework for the verification of conflict detection algorithms. We study the correctness and certainty properties of the AILS Alerting algorithm in Section 3. The last section summarizes our work and contains concluding remarks. Along this paper, and for readability reasons, we have used standard mathematics and traditional logic reasoning. Nevertheless, our development has been formally checked in the general verification system PVS [8]. See [7] for an extended version of this paper. All the PVS theories and proofs are available at `http://shemesh.larc.nasa.gov/fm/ftp/ails/`.

2 Conflict Detection Framework

2.1 Scope

Our framework consists of three elements:

1. a continuous model of aircraft trajectories,
2. the concept of intruder and evader aircraft, and
3. the correctness and certainty properties.

Aircraft Trajectories At the basis of our conflict detection framework is the concept of *aircraft trajectory.* A *trajectory* describes a continuous path in the x-y (horizontal) plane subject to constraints imposed by the aircraft dynamics.[1] It consists of *differentiable* functions in $\mathbb{R} \rightarrow \mathbb{R}$, which map *time* into *heading*, *position*, *bank angle*, and *ground speed* of the aircraft.

We assume two things about the velocity vector of an aircraft. First, an aircraft is moving at constant ground speed v, hence the velocity vector is completely determined by its polar angle θ called the heading of the aircraft. Second, the variation of the heading is proportional to the tangent of the bank angle of the aircraft. Formally, this can be expressed by the equations

$$x'(t) = v\cos(\theta(t)) \tag{1}$$
$$y'(t) = v\sin(\theta(t)) \tag{2}$$
$$\theta'(t) = (g/v)\tan(\phi(t)) \tag{3}$$

where x, y, θ, ϕ are differentiable functions mapping time to location coordinates, heading, and bank angle, respectively. Equations 1 and 2 state that the derivative of the position functions gives the velocity vector of the aircraft. Equation 3 relates the bank angle with the heading of the aircraft. That equation states that the rate of direction change of an aircraft is proportional to the tangent of the bank angle by a factor of g/v, where g is the gravitational force.

The constant ground speed assumption is imposed by the AILS system designers. It is justified by the fact that during AILS operations, aircraft are on final approach and their velocities are restricted.

In addition to the above physical constraints, we impose a maximum bank angle operational constraint for commercial aircraft to be 35^o, i.e.,

$$|\phi(t)| \leq 35\pi/180. \tag{4}$$

Henceforth, we use the constant $\texttt{MaxBank} = 35\pi/180$.

Evader and Intruder Aircraft We assume a pair of aircraft, one labeled *evader* and the other *intruder*. In the AILS system, the evader represents an aircraft flying on normal conditions while the intruder represents a blundering aircraft. The AILS algorithm runs twice on each airplane: the first execution treats the local aircraft as the evader an the foreign aircraft as the intruder; the second execution interchanges the roles of intruder and evader aircraft.

Multiple aircraft scenarios can be modeled as sequential composition of pairwise aircraft conflict detection algorithms. Notice that, in contrast to conflict detection algorithms, conflict *resolution* algorithms for multiple aircraft system are usually not compositional as a pair of aircraft could create new conflicts in previously solved aircraft. Conflict resolution, however, is beyond the scope of this paper.

[1] The vertical separation is typically handled separately. This will be studied in future work.

Equations 1-4 apply to both evader and intruder aircraft. State functions representing the state of the evader and intruder aircraft are subscripted with lowercase letters e and i, respectively. The AILS system assumes that only one aircraft is diverting from its intended landing path. This assumption is also a system designer assumption. It is based on a probabilistic failure assessment. Hence, an additional restriction is imposed on evader trajectories by constraining the bank angle $\phi_e(t) = 0$. This constraint makes the heading of the evader constant and its trajectories straight lines. Without loss of generality, we can chose a coordinate system where the x-axis coincides with the evader trajectory making the heading angle of the evader aircraft always 0. Thus, the equations for the evader can be rephrased and integrated

$$x_e(t) = X_e + v_e t \tag{5}$$
$$y_e(t) = Y_e \tag{6}$$
$$\theta_e(t) = 0 \tag{7}$$
$$\phi_e(t) = 0 \tag{8}$$

where X_e and Y_e are the coordinates of the initial evader position.

Correctness and Certainty Conflict detection is based on the ability to predict future aircraft locations for a given lookahead time $T > 0$. Two aircraft have a *(potential) conflict* at time T, if there exists a trajectory leading to a distance between the aircraft less than a given value `ConflictRange` at time T. Assuming that the aircraft have reliable access to accurate data flight information, two key properties that must be established for a conflict detection algorithm are: (1) any future conflicts within the lookahead time will be detected, and (2) a conflict detection reflects a potential conflict within the lookahead time. The first property is called *correctness* and the latter one is *certainty*. Notice that correctness means that conflicts will not go undetected and certainty means that the algorithm will not detect conflicts that do not exist, possibly leading to false alarms.

Since possible conflicts that are not detected may lead to mid-air collisions, correctness is a much more critical feature, from a safety point of view, than certainty. However, false alarms will have negative effects both on safety and in the overall performance of the airspace system [4].

Predictions of aircraft trajectories are made to determine if a conflict exists in a given lookahead time. Two types of information can be used for prediction: (1) intent information for medium to long lookahead times; and (2) state information for short to medium lookahead times. Intent information refers to information in flight plans, destination, in route way points, etc. State information uses the airplane heading, speed and location to predict future aircraft states. In this paper, we are only concerned with trajectory prediction based on state information.

2.2 Main Lemmas

We have modeled the motion of aircraft by differentiable functions from $\mathbb{R}$ to $\mathbb{R}$. To establish the basic properties of trajectories, we shall need several lemmas that concern differentiable functions, coordinate systems, and objects in motion in general. We present in this section, the main lemmas we have had to prove on such topics.

Elementary Differential Calculus Like most theorem provers, PVS has little automated support for non-linear arithmetic and real analysis. We have extended the pre-defined theory of real numbers and the theory of differential functions developed in [1] with theories dealing with trigonometric and other transcendental functions.

Non-effective real functions are declared in PVS as uninterpreted functions. Their behavior is given axiomatically. For example, cos and sin are functions from reals to the real interval $[-1 \ldots 1]$ satisfying, among other properties, $\sin(a)^2 + \cos(b)^2 = 1$. In a similar way, $\sqrt{\cdot}$ is a function from non-negative reals to non-negative reals such that $\sqrt{a}^2 = a$ for $a \geq 0$. From this axiom, we can prove, for instance, that $\sqrt{a^2} = a$ for $a \geq 0$.

The concept of differentiability and derivative is treated the same way. We use an uninterpreted predicate `Differentiable` over functions mapping real numbers to real numbers and an uninterpreted higher-order function `D` (for derivative) mapping `Differentiable` functions to functions from real numbers to real numbers. Typical axioms assert that sin is differentiable and that its derivative is cos. For our proofs, we have managed to avoid the concept of differentiability over an interval. For instance, the square root function is not assumed to be differentiable, but we have an axiom stating that for all differentiable functions f mapping real numbers to *positive* real numbers, the function $x \mapsto \sqrt{f(x)}$ is differentiable. We do not know if this trick is sufficient in general, or if a general theory of differentiability over an interval is needed in other examples.

Another important axiom defining the concept of derivative is the following theorem of calculus (that can be seen as a consequence of Rolle's theorem).

Theorem 1 (`monotonic_anti_deriv`).

$$\begin{aligned}
&\forall f, g : \mathcal{R} \to \mathcal{R}.\ \forall a, b : \mathcal{R}.\ a \leq b \quad \supset \\
&\quad (\forall c : \mathcal{R}.\ a \leq c \leq b \ \supset\ f'(c) \leq g'(c)) \quad \supset \\
&\quad f(b) - f(a) \leq g(b) - g(a).
\end{aligned}$$

In the verification process it is sometimes necessary to perform calculations on expressions containing non-effective functions such as the trigonometric functions. It is tempting to use approximation series to define, for instance, sin and cos. However, mixing approximation series and axiomatic definitions of trigonometric functions may be a source of paradoxes. Say for example that sin and cos compute approximate values of the real ones. It will be very unlikely that $\sin(a)^2 + \cos(a)^2$ evaluates to 1 for any value of a. In order to avoid that kind of inconsistencies, we mix approximations and uninterpreted functions in a very rigorous way. Assume we want to prove that $e_1[\sin(a)]^+ \leq e_2[\cos(b)]^+$, where $e[s]^+$ stands for a context expression e containing a distinguished positive occurrence of s. Then, we find a computable upper bound of $\sin(a)$, say $\sin_{ub}(a)$, and a computable lower bound of $\cos(b)$, say $\cos_{lb}(b)$. Finally, we prove $e_1[\sin(a)]^+ \leq e_2[\cos(b)]^+$ as follows

$$e_1[\sin(a)]^+ \leq e_1[\sin_{ub}(a)]^+ \tag{9}$$
$$e_1[\sin_{ub}(a)]^+ \leq e_2[\cos_{lb}(b)]^+ \tag{10}$$
$$e_2[\cos_{lb}(b)]^+ \leq e_2[\cos(b)]^+ \tag{11}$$

Most of the times, Formulas 9 and 11 are simple to discharge. If $e_1[\sin(a)_{ub}]^+$ and $e_2[\cos(b)_{lb}]^+$ are computable then we prove Formula 10 by evaluating the expressions. Otherwise, we use the same technique to remove other non-computable values. Eventually, we will get two expressions that we can evaluate. This technique is so used and simple that we have developed PVS strategies to automate the work. As for computable definitions of $\sin_{ub}, \sin_{lb}, \cos_{ub}, \cos_{lb}$, we have used partial approximation by series:

$$\sin_{lb}(a) = \textstyle\sum_{i=1}^{4}(-1)^{i-1}\frac{a^{2i-1}}{(2i-1)!} \qquad \sin_{ub}(a) = \textstyle\sum_{i=1}^{5}(-1)^{i-1}\frac{a^{2i-1}}{(2i-1)!}$$

$$\cos_{lb}(a) = 1 + \textstyle\sum_{i=1}^{3}(-1)^{i}\frac{a^{2i}}{(2i)!} \qquad \cos_{ub}(a) = 1 + \textstyle\sum_{i=1}^{4}(-1)^{i}\frac{a^{2i}}{(2i)!}$$

and the axioms

Axiom 1 (`PI`)

$$3.14 \leq \pi \leq 3.15.$$

Axiom 2 (`SIN`)

$$0 \leq a \leq \pi \supset \sin_{lb}(a) \leq \sin(a) \leq \sin_{ub}(a).$$

Axiom 3 (`COS`)

$$-\pi/2 \leq a \leq \pi/2 \supset \cos_{lb}(a) \leq \cos(a) \leq \cos_{ub}(a).$$

The fact that we have stated many axioms in this section reflect the fact that we have focused on developing proofs of properties of our framework for analysis of conflict detection algorithms using well-known results of calculus, not on developing a calculus library. On a longer term research effort, these uninterpreted constants should be replaced by definitions and these axioms would become theorems.

Geometry An important technique used in our formal development is to take as reference a new system of coordinates where the origin is the position of the evader aircraft at a given time T, i.e., $(x_e(T), y_e(T))$, and the x-y plane has been rotated by $\theta_i(0)$ degrees. We recall that $\theta_i(0)$ is the heading of the intruder aircraft at time 0. The new $\hat{x}$-$\hat{y}$ plane, is defined as follows

$$\hat{x}(t) = \cos(\theta_i(0))[x(t) - x_e(T)] + \sin(\theta_i(0))[y(t) - y_e(T)] \tag{12}$$
$$\hat{y}(t) = \cos(\theta_i(0))[y(t) - y_e(T)] - \sin(\theta_i(0))[x(t) - x_e(T)] \tag{13}$$

We have formally proven several properties related to changes of coordinate systems. For instance, lemma `isometric` states that distances are invariant under rotation and translation of the coordinate system.

Lemma 1 (`isometric`).

$$(x_1(t) - x_2(t))^2 + (y_1(t) - y_2(t))^2 = (\hat{x_1}(t) - \hat{x_2}(t))^2 + (\hat{y_1}(t) - \hat{y_2}(t))^2.$$

Kinematics Let us now turn to the lemmas that involve moving objects. These lemmas were needed to prove safety properties of the AILS algorithm, but they are still rather general and some of them have been reused to prove properties over other air traffic control algorithms.

The first example concerns the ability to determine whether the straight-line trajectories of two aircraft are diverging or converging and to find the point of closest separation of the projected trajectories. This amounts to finding the minimum of the distance between two straight-line trajectories. If the evader aircraft is assumed to have heading 0 and the intruder aircraft has heading θ, then the equations defining the *projected trajectories* are

$$\begin{aligned}
x_e(t) &= x_e(0) + v_e t \\
y_e(t) &= y_e(0) \\
x_i(t) &= x_i(0) + v_i t \cos(\theta) \\
y_i(t) &= y_i(0) + v_i t \sin(\theta)
\end{aligned}$$

and the distance between the *projected trajectories* at time t, $R(t)$, can be computed as follows:

$$\begin{aligned}
\Delta_x(t) &= x_i(t) - x_e(t) \\
\Delta_y(t) &= y_i(t) - y_e(t)
\end{aligned}$$

$$R(t) = \sqrt{\Delta_x(t)^2 + \Delta_y(t)^2} \tag{14}$$

To find the minimum of $R(t)$, first the derivative of $R(t)$ is computed:

$$R'(t) = \frac{\Delta_x(t)\Delta'_x + \Delta_y(t)\Delta'_y}{R(t)}$$

where

$$\Delta'_x = v_i \cos(\theta) - v_e$$
$$\Delta'_y = v_i \sin(\theta)$$

We have formally verified that when $R'(t+\tau) = 0$, the time τ, relative to t, is the time of closest separation between the aircraft. The solution to this equation is:

$$\tau(t) = -\frac{\Delta_x(t)\Delta'_x + \Delta_y(t)\Delta'_y}{{\Delta'_x}^2 + {\Delta'_y}^2} \tag{15}$$

It is important to note that τ is undefined, i.e., denominator is zero, when the aircraft are parallel and the ground speeds are equal.

For any time t, if $\tau(t)$ is negative or zero, the tracks are diverging or parallel, respectively. If $\tau(t)$ is greater than zero, the tracks are converging and $\tau(t)$ is the time of closest separation relative to t. In PVS:

Lemma 2 (`derivative_eq_zero_min`).

$$R(t_1 + \tau(t_1)) \leq R(t_1 + t_2).$$

Lemma 3 (`asymptotic_decrease_tau`).

$$t_1 \leq t_2 \leq \tau(t) \supset R(t + t_1) \geq R(t + t_2).$$

Lemma 4 (`asymptotic_increase_tau`).

$$\tau(t) \leq t_1 \leq t_2 \supset R(t + t_1) \leq R(t + t_2).$$

A second example concerns the maximum and minimum distance traveled by an aircraft in a given time whose speed is constant and bank angle bounded. In particular, vt is the farthest distance, i.e., via straight line, that can be reached by an aircraft moving at constant speed v in t seconds. That property is called `YCNGFTYS`, which stands for *You Cannot Go Faster Than Your Speed*, and it has been formally verified in PVS.

Theorem 2 (`YCNGFTYS`).

$$t \geq 0 \supset \sqrt{(x(t) - x(0))^2 + (y(t) - y(0))^2} \leq vt.$$

For an aircraft moving at constant speed v and with a constant bank angle ϕ, Figure 1, the distance from the position at time 0 to the position at time t is given by the formula

$$m(v, \phi, t) = 2r(v, \phi) \sin(vt/2r(v, \phi)) \tag{16}$$

where $r(v, \phi)$ is the turn radius of the aircraft.

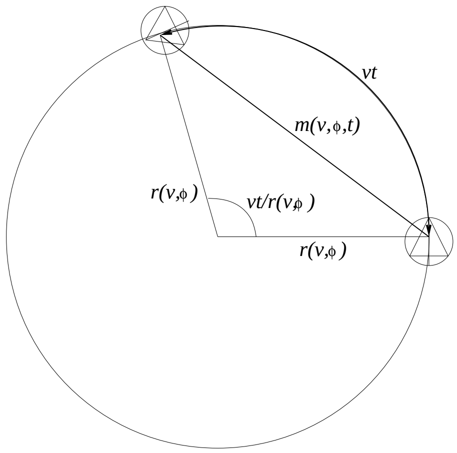

Fig. 1. Distance Traveled in Curved Trajectory

The turn radius $r(v, \phi)$ can be calculated as follows.

$$\begin{aligned} vt/r(\phi, v) &= (g/v)\tan(\phi)t \text{ (From Equation 3)} \\ v/r(v, \phi) &= (g/v)\tan(\phi) \text{ (Simplifying } t\text{)}. \end{aligned}$$

Thus,

$$r(v, \phi) = v^2/(g\tan(\phi)). \tag{17}$$

According to Formula 4, the maximum change of heading per second of an aircraft moving at constant speed v is given by

$$\rho(v) = (g/v)\tan(\mathtt{MaxBank}). \tag{18}$$

From Equation 17 and Equation 18:

$$r(v, \mathtt{MaxBank}) = v/\rho(v) \tag{19}$$

and from Equation 16 and Equation 19:

$$m(v, \mathtt{MaxBank}, t) = 2r(v, \mathtt{MaxBank})\sin(\rho(v)t/2). \tag{20}$$

When $0 \leq \rho(v)t \leq 2$, we have formally verified in PVS that $m(v, \mathtt{MaxBank}, t)$ is the minimum distance traveled by an aircraft moving at constant speed v in t seconds[2]. The property is called `YCNGSTYS`, which stands for *You Cannot Go Slower Than Your Speed.*

[2] We conjecture that the property still holds for $0 \leq \rho(v)t \leq 2\pi$; but, we could not find a formal proof of this proposition.

Theorem 3 (YCNGSTYS).

$$0 \le \rho(v)t \le 2 \supset m(v, \texttt{MaxBank}, t) \le \sqrt{(x(t) - x(0))^2 + (y(t) - y(0))^2}.$$

According to theorems YCNGFTYS (Theorem 2) and YCNGSTYS (Theorem 3), for an aircraft moving at constant ground speed v, the inner circle of radius $m(v, \texttt{MaxBank}, t)$ and the outer circle of radius vt, both centered at the current position of the aircraft, delimit the area that could be reached by the aircraft at time t.

3 Verification of the AILS Alerting Algorithm

The AILS system is intended to enable independent parallel landings to closely spaced parallel runways during reduced visibility conditions. An integral part of the system is the alerting algorithm which provides a sequence of alarms when the aircraft are diverting from their intended landing paths.

The original AILS algorithm was written in FORTRAN at Langley Research Center. It has been revised several times and the latest version, flown in the Boeing 757 experimental aircraft, was written by Honeywell. Since the AILS alerting system is a multilevel alarm system, it is possible to prove properties of the different levels of alarms. The alarm we are most concerned with is the alarm the evader aircraft should receive when the intruder aircraft is threatening its airspace. This is the alarm we will be discussing in the remaining of this section.

We have specified that algorithm in PVS by a predicate `ails_alert` that takes the initial measured states of an intruder aircraft and an evader aircraft, and returns `true` or `false` depending on whether the alarm is issued or not. The specification of the algorithm is the translation of the original FORTRAN code to a higher-order logic description. The translation was almost one-to-one with the exception of replacing DO-LOOPS statements with recursive definitions.

3.1 Correctness

The verification of the alerting algorithm consists of showing that for all trajectories that could lead to a conflict within a given time T, the alerting algorithm will issue an alarm. We called this property the correctness of the AILS Alerting Algorithm. Using the trajectory framework described in previous sections and the model of the algorithm, we formulated the properties necessary for the verification.

We started the verification process by trying to show correctness for a value of T equals 19 seconds which is the lookahead time for the algorithm. Soon, we discovered that the correctness property was not provable for $T = 19s$. Indeed, a counter example for $T = 10.5$ seconds was found. The counter example says that there exists a trajectory which will bring the two aircraft within 10.5 seconds of a conflict without an alarm being issued. This counter example is in some sense minimal, as we have been able to prove correctness for $T \le 10$ seconds and the time step between two consecutive calls of the algorithm is 0.5 seconds.

To prove the correctness property for $T \le 10$, it is sufficient to prove that if a conflict is going to occur in a time ranging between $9.5s$ and $10s$ then an alarm is raised. Note that if a conflict is going to occur in less that $9.5s$ an alarm has already been raised[3]. Hence, we have the following correctness theorem.

Theorem 4 (`ails_correctness`).

$$\forall i, e.\ 9.5 \le T \le 10\ \wedge\ \mathit{conflict}_{ie}(T) \quad \supset$$
$$\mathit{ails\_alert}(\mathit{measure2state}(i, 0), \mathit{measure2state}(e, 0))$$

where `measure2state`(a, t) is the state of the aircraft a at time t and the proposition `conflict`$_{ie}(T)$ means that there exists a potential conflict between the aircraft at time T.

The proof of this theorem uses the notion of state of an aircraft consisting of its location, heading, and bank angle. The states of the aircraft are part of the geometrical description of the problem, but also are the data processed by the AILS algorithm. Thus, they constitute the interface between the algorithm and its physical environment. In this paper, we have assumed that these measurements are made without error. Although with ADS-B exchange of information the errors can be made very small, it should be included in future work.

The proof proceeds by proving independently that (1) if a conflict is to occur in a time ranging between $9.5s$ and $10s$, then the pair of aircraft is in a certain region $\mathcal{G}$ of the state space; and (2) if it is in this region $\mathcal{G}$, then an alarm is issued. The proof of (1) involves only the geometry and the kinematics of the physical environment, while the second proof concerns only the specification of the algorithm. The region $\mathcal{G}$ constitutes the interface between the two proofs, in the same way as the measured state is the interface between the algorithm and its physical environment.

More precisely, the first lemma proves that if a conflict is to occur, then

1. the distance of the initial position of the intruder to the final position of the evader ranges between a distance `MinDistance` and `MaxDistance` that are the minimum and the maximum distance the intruder may run in a time ranging between $9.5s$ and $10s$ (as determined by theorems 2 and 3),
2. the initial heading of the intruder also ranges between two bounds,
3. $\tau(0) > 0$, i.e., the aircraft are converging at current time.

In this last case, we have to prove that that if the aircraft are diverging at current time, although the intruder can change heading, it cannot do it sufficiently fast to create a conflict. The second lemma proves that the AILS algorithm issues an alarm if all these conditions hold.

It is important to note that what we have shown for $9.5 \le T \le 10$ is a worst case scenario. In simulations with actively flying airline pilots, it was found that the pilot reaction time was in average approximately 2 seconds. Also, according to the logic of the AILS alerting system, the intruder aircraft should receive two

[3] Due to operational constraints, when the AILS system is first engaged during a final approach, there is a safe window of at least 9.5 seconds where no conflict can occur.

path deviation indications and one traffic caution indication before the evader aircraft is alerted.

3.2 Uncertainty

We have also proven that although the AILS algorithm is correct, it is not certain, i.e., it may issue false alarms.

Theorem 5 (`ails_uncertainty`).

$$\begin{array}{c}\exists s_i, s_e : \mathit{State}.\ \forall i, e.\ 0 < T \leq 10 \\ s_i = \mathit{measure2state}(i, 0)\ \wedge\ s_e = \mathit{measure2state}(e, 0) \quad \supset \\ \mathit{ails\_alert}(s_i, s_e)\ \wedge\ \neg \mathit{conflict}_{ie}(T).\end{array}$$

To prove this theorem, we simply find states of intruder and evader aircraft that issue an alarm, but where the aircraft cannot conflict within 10 seconds.

4 Conclusion

In this paper, we have presented the foundation for a new approach to verifying the safety of conflict detection algorithms that may one day be deployed in the national airspace. Such algorithms are an enabling technology for free flight, where pilots are allowed to fly their own preferred trajectories. The introduction of these algorithms in a free-flight context raises significant safety issues. Historically the trajectories of aircraft have been managed by ground controllers through use of aircraft position data obtained from radar. Under this approach, the primary responsibility for maintaining aircraft separation has been borne by the air traffic controller. But under a free-flight approach, much of the responsibility for maintaining separation will be transferred to the pilots *and the software which provides them aircraft positions*, i.e., via Cockpit Display of Traffic Information (CDTI), and warnings of potential conflicts. We believe that current methods for gaining assurance about the safety of ground-based decision-aid software are inadequate for many of the software systems that will be deployed in the future in support of free flight. The current approach is based upon human-factors experimentation using high fidelity simulations. In the current approach, where the responsibility for safety resides in the human controller, this is clearly the right approach. The primary question to be answered is whether the software provides the controllers with useful information that aids them in their decision making. But as software takes on more and more of the responsibility for generating aircraft trajectories and detecting potential conflicts and perhaps even producing (and executing?) the evasive maneuvers, we will need additional tools to guarantee safety. It is our view that it is essential that the correctness of the algorithm be established for *all possible* situations. Simulation and testing cannot accomplish this. Although simulation and controlled experimentation are clearly necessary, they are not sufficient to guarantee safety. This can only be done by analytical means, i.e., formal verification. We should also note that it will also be necessary to demonstrate that the implementation of these algorithms in software is correct. This refinement verification, in our view, must also be accomplished using formal methods. We hope to explore this issue with our colleagues in future work.

The trajectory model used in this paper is the result of investigating different approaches. Earlier work looked at more discrete versions with the expectation that this would lead to a more tractable verification task. Unfortunately the discretization of the trajectories led to significant (and accumulating) modeling error that led to erroneous conclusions. In the end, we have settled on modeling trajectories as differentiable functions over real numbers. These trajectories are constrained by the dynamics of an aircraft. These constraints enable one to establish high level properties that delineate when a conflict is possible. In this paper we have developed a formal theory about trajectories that can serve as the basis for the formal analysis of conflict detection and resolution (CD&R) algorithms. There are several limitations to this formal theory that will be addressed in future work. These include: (1) the theory only deals with 2 aircraft, (2) the vertical dimension is not modeled, and (3) aircraft data measurement errors are not modeled.

Because the trajectories of the aircraft are modeled by differentiable functions over real numbers and the discrete algorithms are periodically executed on a digital computer, this problem domain falls into the domain of hybrid models. The hybrid nature of this domain makes the verification problem especially difficult. Automatic methods such as model checking cannot directly handle the continuous trajectories, i.e., infinite state space, and discretization leads to unacceptable errors. We are forced to reason about such systems in the context of a fully general theorem prover designed to handle a rich logic such as higher-order logic, type theory, or ZFC set theory. We have used the PVS theorem prover in our work and found this tool to be sufficient to handle this problem domain, but our work was often impeded by PVS's baroque method for dealing with non-linear arithmetic. Although PVS provides a suite of decision procedures that automate much of the tedium of theorem prover, in this arena, one must wrestle with the prover in order to make progress. Adding capability for reasoning about formulas containing non-linear arithmetic in theorem provers is a current area of research.

Future work will concentrate on applying this modeling framework to specific CD&R algorithms and perhaps to self-spacing and merging algorithms designed to increase capacity in the terminal area. We would also like to develop formal methods for analyzing conflict resolution schemes and the safety of algorithmically-generated evasive maneuvers [3]. The CD&R methods must be generalized to cover sets of aircraft constrained by formally specified notions of aircraft density (static or dynamic). We would also like to generalize the methods to encompass measurement error and data errors. This is a necessary step toward developing formal methods useful for the design and implementation phases of realistic avionics.

Acknowledgment The authors would like to thank Alfons Geser, Michael Holloway, and the anonymous referees for their helpful comments on preliminary versions of this paper.

References

1. B. Dutertre. Elements of mathematical analysis in PVS. In J. Von Wright, J. Grundy, and J. Harrison, editors, *Ninth international Conference on Theorem Proving in Higher Order Logics TPHOL*, volume 1125 of *Lecture Notes in Computer Science*, pages 141–156, Turku, Finland, August 1996. Springer Verlag.
2. V. Carreño and C. Muñoz. Aircraft trajectory modeling and alerting algorithm verification. In J. Harrison and M. Aagaard, editors, *Theorem Proving in Higher Order Logics: 13th International Conference, TPHOLs 2000*, volume 1869 of *Lecture Notes in Computer Science*, pages 90–105. Springer-Verlag, 2000. An earlier version appears as report NASA/CR-2000-210097 ICASE No. 2000-16.
3. G. Dowek, C. Muñoz, and A. Geser. Tactical conflict detection and resolution in a 3-D airspace. Technical Report NASA/CR-2001-210853 ICASE Report No. 2001-7, ICASE-NASA Langley, ICASE Mail Stop 132C, NASA Langley Research Center, Hampton VA 23681-2199, USA, April 2001.
4. J. Kuchar and Jr. R. Hansman. A unified methodology for the evaluation of hazard alerting systems. Technical Report ASL-95-1, ASL MIT Aeronautical System Laboratory, January 1995.
5. J. Kuchar and L. Yang. Survey of conflict detection and resolution modeling methods. In *AIAA Guidance, Navigation, and Control Conference*, volume AIAA-97-3732, pages 1388–1397, New Orleans, LA, August 1997.
6. J. Lygeros and N. Lynch. On the formal verification of the TCAS conflict resolution algorithms. In *Proceedings 36th IEEE Conference on Decision and Control,* San Diego, CA, pages 1829–1834, December 1997. Extended abstract.
7. C. Muñoz, R.W. Butler, V. Carreño, and G. Dowek. On the verification of conflict detection algorithms. Technical Report NASA/TM-2001-210864, NASA Langley Research Center, NASA LaRC Hampton VA 23681-2199, USA, May 2001.
8. S. Owre, J. M. Rushby, and N. Shankar. PVS: A prototype verification system. In Deepak Kapur, editor, *11th International Conference on Automated Deduction (CADE)*, volume 607 of *Lecture Notes in Artificial Intelligence*, pages 748–752, Saratoga, NY, June 1992. Springer-Verlag.
9. Radio Technical Commission for Aeronautics. Final report of the RTCA board of directors' select committee on free flight. Technical Report Issued 1-18-95, RTCA, Washington, DC, 1995.
10. L. Rine, T. Abbott, G. Lohr, D. Elliott, M. Waller, and R. Perry. The flight deck perspective of the NASA Langley AILS concept. Technical Report NASA/TM-2000-209841, NASA, January 2000.
11. C. Tomlin, G. Pappas, and S. Sastry. Conflict resolution for air traffic management: A study in multi-agent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4), April 1998.