A GEOMETRIC APPROACH TO STRATEGIC CONFLICT DETECTION AND RESOLUTION[1]

Alfons Geser, César Muñoz[2], ICASE - NASA LaRC, Hampton, Virginia

Abstract

Conflict detection and resolution (CD&R) systems predict loss of separation between aircraft and propose conflict avoidance maneuvers for the aircraft involved in the conflict. Given a pair of aircraft in conflict, the ownship and the intruder, the resolution system diverts the ownship from its original trajectory. In this paper, we introduce the concept of *recovery course*. A recovery course redirects the ownship to its original target waypoint without introducing new conflicts, in a geometric optimal way. Based on resolution and recovery concepts, we also develop a strategic CD&R approach that produces conflict-free flight plans for the ownship and the intruder aircraft. The resolution and recovery algorithm is computationally efficient and amenable to formal verification. We provide a rigorous analysis of the problem and the basis of the correctness proof of our approach.

Introduction

Conflict Detection and Resolution (CD&R) systems are designed to warn air traffic controllers or pilots about an imminent loss of separation between aircraft, and to assist them in a corrective maneuver. Algorithms for CD&R have been widely studied over the last decade and several have been reported in the literature (for a survey on CD&R methods see [1]). Emerging reliable surveillance and communication technologies enable *airborne* CD&R capabilities, and so, they are becoming fundamental features in new concepts for air traffic management, such as Free-Flight [2] and Distributed Air-Ground Traffic Management [3]. These concepts address the expected increase in air traffic density in the next decades, by distributing among the different actors of the airspace system the responsibility for keeping minimum traffic separation.

In contrast to ground-based systems, airborne systems have a limited access to computational resources and are distributed. To target the complexity of a free-flight environment, new approaches for CD&R have been proposed based on non-standard programming techniques such as genetic algorithms [4,5,6], neural networks [7], game theory [8], graph theory [9], or semidefinite programming [10]. These approaches deal with issues such as multiple aircraft conflicts and uncertainties in the prediction of aircraft trajectories. Given the computational complexity of some of these approaches, they usually require time and space discretizations.

A more classical approach to CD&R is the so-called *geometric* approach [11,12,13,14]. In this approach, aircraft trajectory predictions are based on linear projections of current aircraft states. Linear projections can be computed efficiently and, moreover, prediction errors are negligible during short look-ahead times. For the latter reason, this approach is also referred as *tactical*. For large look-ahead times a more *strategic* approach that looks at the pilot *intent information*, i.e., the flight plan, is in order. While tactical approaches have well-understood geometric descriptions that allow for efficient and clear algorithms, they fall short on pilots' expectations in some field studies [15]. Strategic approaches seem to be more appreciated by pilots, but their theory is far less understood.

In a previous work [14], we have proposed a geometric optimization algorithm, called KB3D, for CD&R in a 3-D airspace. In this paper, we address the *recovery* problem, that is, redirecting the ownship to the original path while maintaining the minimum required separation at all times. The new proposed algorithm is on the border between tactical and strategic algorithms. Its inputs are the

[1] This work was supported by the National Aeronautics and Space Administration under NASA Contract No. NAS1-97046 while the authors were in residence at ICASE, NASA Langley Research Center, Hampton, VA 23681-2199, USA.

[2] Corresponding author: munoz@icase.edu, ICASE, Mail Stop 132C, NASA Langley Research Center, Hampton, VA 23681-2199, USA.

three-dimensional position and velocity vectors of the ownship and intruder aircraft, and the time where the ownship is required to arrive at its target point ("Required Time of Arrival", RTA). The RTA defines the position of the target point by the ownship's constant movement. RTA is a limited form of intent information: the target point may be the next trajectory change point in the ownship flight plan. Assuming a loss of separation between two aircraft, the algorithm outputs a choice of maneuvers for the ownship. Each maneuver consists of an *escape course* that brings the ownship off the predicted conflict, and a subsequent *recovery course* that leads the ownship back to its original target waypoint. If the ownship follows any proposed maneuver then it arrives at RTA at the scheduled position without having experienced a loss of separation at any time. In the meantime, the intruder aircraft is assumed to continue its current trajectory. That is, ownship's resolution-recovery maneuvers take place assuming no cooperation of the intruder aircraft.

The various maneuvers that our algorithm proposes differ in the constraints they satisfy. For example, one constraint requires that during the maneuver the ownship may only change its ground speed, but not its heading or its vertical speed. Another constraint requires that only the vertical speed of the ownship may change. Imposing such constraints has a number of benefits:

- It restricts the number of choices to finitely many.
- It simplifies the calculations performed by the algorithm.
- It is simple to conceive and to perform by the crew.
- It enhances passenger comfort.

The maneuvers that our algorithm outputs may be rendered at a display for air traffic controllers or pilots who may select among the proposed solutions. Our algorithm is also suitable for use underneath a trajectory planner that may perform the selection. Indeed, we describe a strategic CD&R approach that produces conflict-free flight plans, based on our resolution and recovery algorithm.

However, we do not address the question of physical feasibility of the maneuvers proposed by the algorithm. In particular, the algorithm does not check for minimum/maximum altitude/airspeed. Nor does it implement cost-based analysis such as fuel consumption. All these analyses require performance data that are not available to the algorithm. They can be implemented in an external module.

The algorithm is computationally efficient, suitable for embedding in a flight-deck computer, and appropriate for formal verification. We provide a rigorous mathematical description of the problem and show that the algorithm is correct, i.e., no matter which of the proposed maneuvers the ownship picks, it will arrive at the target point at the scheduled time while maintaining the minimum required separation to the intruder at all times. Mechanically checked proofs are currently under development. We strongly believe that given the critical nature of CD&R, rigorous techniques and well-understood mathematical models are required to guarantee the overall safety of the new, and more autonomous, air traffic systems.

Definition of the Problem

For conflict detection purposes, aircraft are assumed to be surrounded by an *avoidance region*, which is typically a cylinder of 5 nautical miles of diameter and 1000 feet of height. Two aircraft are said to be in *conflict* when their avoidance regions overlap. In this paper, we take an alternative, but equivalent view, where aircraft are surrounded by *protected zones* twice as big as the individual avoidance regions. In this view, a conflict is the incursion of one aircraft in the protected zone of another one.[3]

We assume the airspace given as a three-dimensional Cartesian coordinate system, where the *z*-axis points upward in the vertical direction. The ownship's initial position, i.e., its position at time $t = 0$, is given by the vector

$\boldsymbol{s_o} = (s_{ox}, s_{oy}, s_{oz})$.

The ownship's original velocity vector is given by

$\boldsymbol{v_o} = (v_{ox}, v_{oy}, v_{oz})$.

Likewise the intruder's initial position $\boldsymbol{s_i}$ and the intruder's velocity vector $\boldsymbol{v_i}$ are given. It is convenient to consider the ownship's motion

[3] In this paper we use *Italic* letters to denote variables. Positions are named '*s*' and velocities '*v*'. Ownship variables are subscripted with '*o*' and intruder variables with '*i*'. Coordinate names '*x*','*y*','*z*' are appended as a subscript to the name. Names on ***boldface*** indicate vector variables.

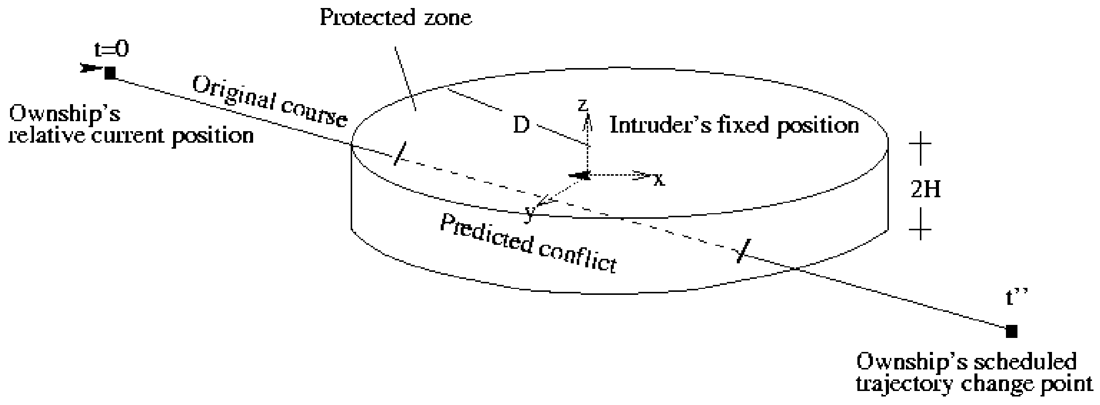
Protected zone
t=0
Original course
Ownship's relative current position
D
z
Intruder's fixed position
x
y
2H
Predicted conflict
t''
Ownship's scheduled trajectory change point

Figure 1: Predicted Conflict

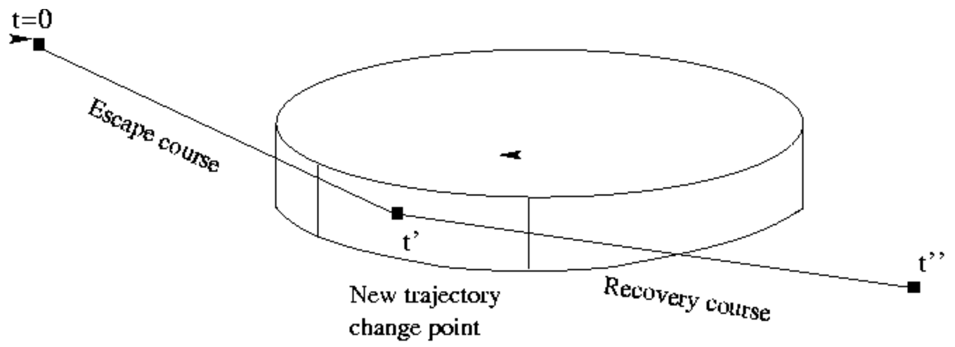

Figure 2: Escape and Recovery Courses

relative to the intruder. For this purpose, we introduce a relative coordinate system where the intruder's position is fixed at the origin, and we consider the relative position vector

$\boldsymbol{s} = (s_x,s_y,s_z) = \boldsymbol{s_o} - \boldsymbol{s_i}$,

and the relative velocity vector

$\boldsymbol{v} = (v_x,v_y,v_z) = \boldsymbol{v_o} - \boldsymbol{v_i}$.

In this coordinate system, the protected zone is defined as a cylinder P around the intruder:

$P = \{(x,y,z) \mid x^2+y^2 < D^2 \text{ and } |z| < H\}$,

where D and H denote the diameter and height of the avoidance region, respectively. The aircraft are said to be *in conflict at time t* when $\boldsymbol{s}+t\boldsymbol{v} \in P$. They are in *predicted conflict* if they are in conflict at some time $0 < t$.

Given a velocity vector $\boldsymbol{v} = (v_x,v_y,v_z)$, we define the following concepts.

- *Ground speed:* Length of the horizontal projection of $\boldsymbol{v}$, i.e., $\sqrt{v_x^2+v_y^2}$.
- *Vertical speed:* Vertical component of $\boldsymbol{v}$, i.e., v_z.
- *Heading:* Direction of the horizontal projection of $\boldsymbol{v}$, i.e., angle α such that:

  $v_x = v\cos(\alpha)$ and $v_y = v\sin(\alpha)$,

  where v is the ground speed of $\boldsymbol{v}$. We avoid explicit references to α in our analytical description of the escape-recovery maneuvers.

The task of the resolution and recovery algorithm (RR3D) is defined as follows:

Inputs

- Initial ownship's relative position $\boldsymbol{s}$.
- Absolute velocity vectors $\boldsymbol{v_o}$ and $\boldsymbol{v_i}$ of ownship and intruder aircraft, respectively.
- Required Time of Arrival (RTA) or target time $t'' > 0$, which determines the target point

  $\boldsymbol{s}'' = \boldsymbol{s}+t''\boldsymbol{v}$.

Assumptions

- *Courses*, i.e., trajectories between waypoints, are line segments. Hence, courses are described by a position, a velocity vector, and a time interval.
- Changes of course or speed are implemented in zero time by an aircraft.

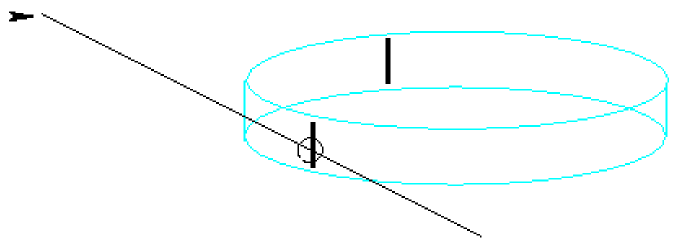

Figure 3: Line Course

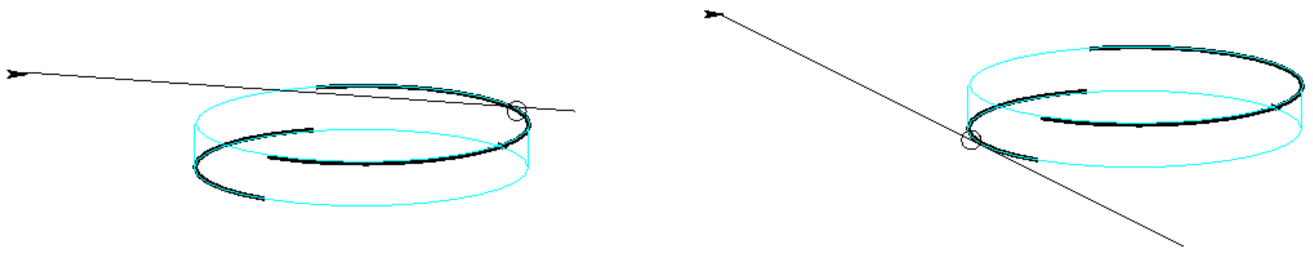

Figure 4: Circle Courses

- Neither at initial time nor at target time the aircraft are in conflict.
- Neither at initial time nor at target time is the ownship at the boundary of the intruder's protected zone.
- The aircraft are in predicted conflict before *t*", i.e., $\boldsymbol{s}+t\boldsymbol{v} \in P$ for some time $0 < t < t''$.

Outputs

A list of *maneuvers* each one a triple $(t', \boldsymbol{v_o}', \boldsymbol{v_o}'')$ composed of

- A *time of switch t'* such that $0 < t' < t''$.
- An *escape velocity vector* $\boldsymbol{v_o}'$ that determines a conflict-free *escape course* for the ownship. A *recovery velocity vector* $\boldsymbol{v_o}''$ that determines a conflict-free and on-time *recovery course* for the ownship.

Figure 1 illustrates the original situation where the aircraft are in conflict during the dashed line. Figure 2 shows a possible pair of escape and recovery courses. The two vertical lines in the protected zone mark the points where the owhship touches *P*. If the ownship flies the escape course from time 0 to *t*', and the recovery course from time *t*' to *t*", then (1) it shall not be in conflict at any time between 0 and *t*", and (2) it shall arrive at $\boldsymbol{s}''$ at time *t*". Note that we assume no cooperation from the intruder aircraft, i.e., we assume that the intruder does not maneuver. Henceforth, we call the ownship's change of the velocity vector from $\boldsymbol{v_o}$ to $\boldsymbol{v_o}'$ the *escape step*, and its change from $\boldsymbol{v_o}'$ to $\boldsymbol{v_o}''$ the *recovery step*.

The ownship's maneuvers shall be constrained in such a way that both $\boldsymbol{v_o}'$ and $\boldsymbol{v_o}''$ satisfy one of the following conditions:

1. *Change of vertical speed only*. The ownship's vertical speed may change but neither its heading nor its ground speed.
2. *Change of ground speed only*. The ownship's ground speed may change but neither its heading nor its vertical speed.
3. *Change of heading*. In the two dimensional projection, the escape course and the recovery course (each in absolute coordinates) form a triangle. By the triangle inequality, the escape course and the recovery course together are longer than the original course. To arrive at the target point at time *t*", the ownship has to compensate the longer way by a greater average ground speed as opposed to its original ground speed. Hence, maneuvers where only heading changes are allowed cannot reach the target point in time. In this case, we propose a change of heading combined with a change of ground speed at time *t*'. For the escape step, the

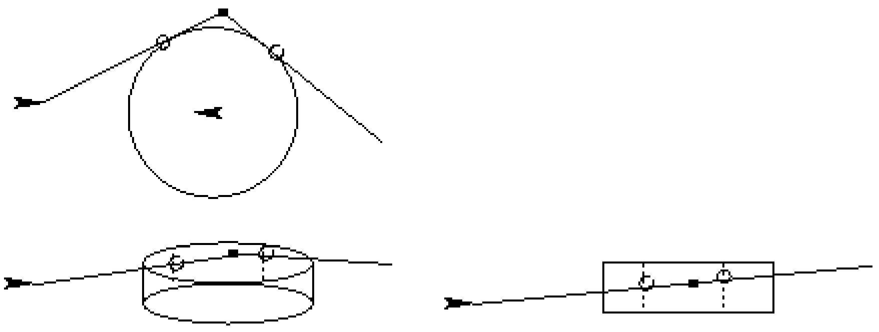
Figure 5: Line/Line (Top View, Perspective View, and Side View)

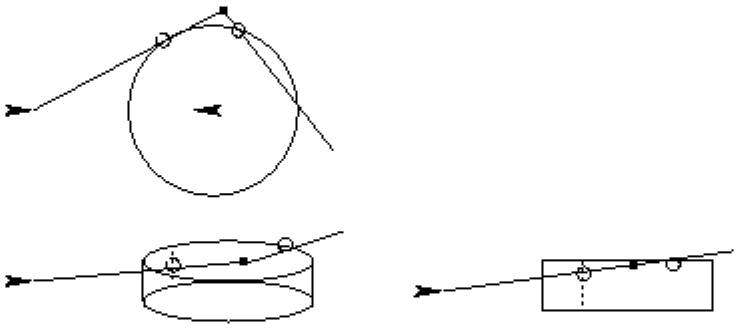
Figure 6: Line/Circle (Top view, Perspective View, and Side View)

ownship's heading may change, but neither its ground speed nor its vertical speed; for the recovery step in addition to a heading change, one must allow for a change of ground speed as well.

Furthermore, in all cases, we require the escape and recovery maneuvers to be tangential to the lateral surface of the protected zone. Together with the constraints, this ensures finiteness of the set of solutions. We distinguish two kinds of tangential escape and recovery courses

- **Line Courses** (Figure 3): Courses that are tangential to the lateral surface of the protected zone.
- **Circle Courses** (Figure 4): Courses that are incident to one of the disks of the protected zone without intersecting its interior.

The bold lines in Figures 3 and 4 indicate the set of points in the protected zone that define the direction of line and circle courses, respectively.

In [16], we provide a rigorous mathematical proof that line and circle cases are *correct*, i.e., they describe conflict-free courses. We just remark here that in a line course, horizontal separation is guaranteed for all times, and that in a circle course, there is a time *t* such that either vertical separation is guaranteed before *t* and horizontal separation is guaranteed after *t*, or vice versa.

Since escape and recovery courses touch *P*, we also conjecture that these kinds of courses are *optimal*, i.e., they avoid conflicts with a minimal change to the original ownship velocity vector $\boldsymbol{v_o}$.

Resolution and Recovery Algorithm

- In this section we develop the algorithm RR3D for 3-D conflict resolution and recovery. We present the algorithm as a set of formulas that describe escape-recovery maneuvers, i.e., triples (t',$\boldsymbol{v_o}'$,$\boldsymbol{v_o}''$) where t' is a time of switch, $\boldsymbol{v_o}'$ is a velocity vector that determines an escape course, and $\boldsymbol{v_o}''$ is a velocity vector that determines a recovery course. The formulas are organized according to the constraints that are imposed on the escape and recovery maneuvers. As explained above, we consider three constraints: change of vertical speed only, change of ground speed only, and change of

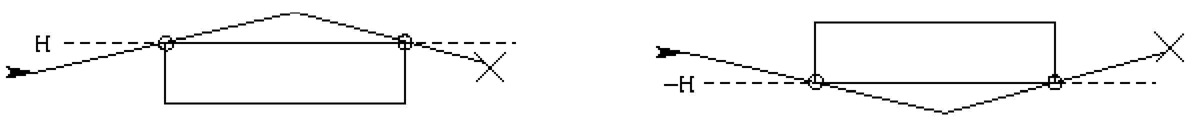

Figure 7: One-Circle Cases (Side Views)

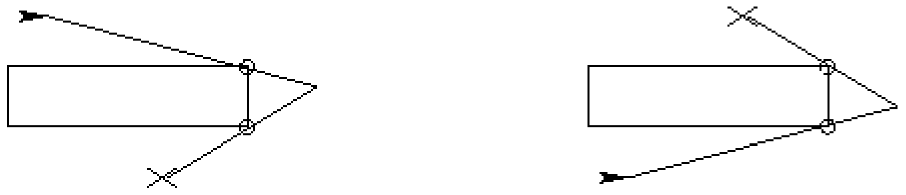

Figure 8: Circle/Circle Cases (Side Views)

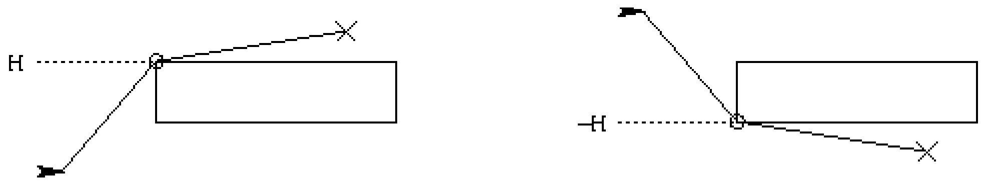

Figure 9: Escape-Circle Cases (Side Views)

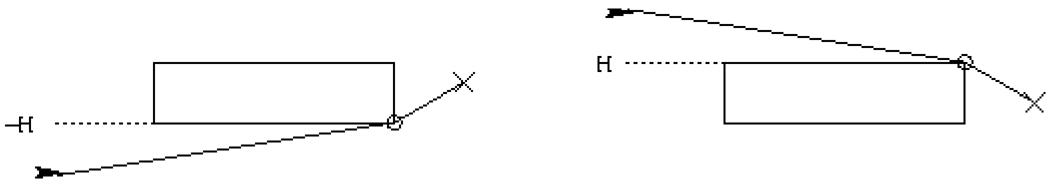

Figure 10: Recovery-Circle Cases (Side Views)

heading combined with a change of ground speed at time *t'*. For each one of the constraints, we distinguish several cases according to the part of the surface of *P* that is touched during the escape and recovery courses. We identify the following cases:

- **Line/Line** (Figure 5): Both escape and recovery courses are line courses.

  Line/Circle (Figure 6): The escape course is a line course and recovery course is a circle course.
- **Circle/Line**: The escape course is a circle course and the recovery course is a line course. This case is symmetric to the line/circle case.
- **One-Circle** (Figure 7): Both escape and recovery courses are circle courses. The circle that is touched in both courses is the same.
- **Circle/Circle** (Figure 8): Both escape and recovery courses are circle courses. The circles that are touched in both courses are different.
- **Escape-Circle** (Figure 9): Both escape and recovery courses are circle courses. The circle that is touched in both cases is the same and it is touched only once. Horizontal separation is guaranteed for the escape course.
- **Recovery-Circle** (Figure 10). Both escape and recovery courses are circle courses. The circle that is touched in both cases is the same and it is touched only once. Horizontal separation is guaranteed for the recovery course.

We note that not all the cases are possible in all situations. For instance, a one-circle case is only possible when a vertical speed change is allowed.

For each one of these cases we use the following approach:

1. We find candidates to maneuvers by solving the equations given by the constraints and the case analysis.
2. We disregard candidates that contradict the assumptions. This may require *sanity checks* with the implicit meaning that only solutions satisfying them are considered any further.
3. Candidates that survive each check are the maneuvers returned by the algorithm. This way, correctness is guaranteed by construction.

We have constructed closed-term solutions for each constraint. For lack of space, we illustrate at a superficial level how the resolution and recovery problem is solved under the constraint “change of heading”. For technical details on all the cases, the reader is encouraged to consult the technical report [16].

Change of Heading

We impose the constraint that for the escape step only the heading of the velocity vector may change. For the recovery step the heading and the horizontal speed may change. Formally,

$v_{ox}'^2+v_{oy}'^2 = v_{ox}^2+v_{oy}^2$, and $v_{oz} = v_{oz}' = v_{oz}''$ (1)

Since the vertical speed is unchanged by the maneuver, we may find the times when the onwship reaches altitudes $\pm H$. If $v_z = 0$ then there are two solutions, $\theta'<\theta''$, to the equation in t: $|s_z+tv_z| = H$. The time t' of switch from the escape course to the recovery course satisfies

$t'(\boldsymbol{v'}-\boldsymbol{v''}) = t''(\boldsymbol{v}-\boldsymbol{v''})$.

This vector equation is the same as a system of equations in each coordinate. With the equation in the *z*-coordinate useless, we have two equations in three unknowns t', v_x'', and v_y''.

For the cases involving an escape (recovery) line course, we assume that $\boldsymbol{s}$ ($\boldsymbol{s''}$) is outside the infinite cylinder, i.e.,

$s_x^2 + s_y^2 > D^2$ ($s_x''^2 + s_y''^2 > D^2$).

Furthermore, for the cases involving a circle course, we assume that relative vertical speed is not zero, i.e., $v_z \neq 0$. Otherwise, there is no solution. We have the following independent solutions.

- **Line/Line.** The situation is shown in Figure 5. We ignore the *z*-coordinate. First, we find the escape velocity vector $\boldsymbol{v'}$, then, the time of switch t', and finally, the recovery velocity vector $\boldsymbol{v''}$.

  For the escape step, we construct the tangent from the point $\boldsymbol{s}$ to the infinite cylinder. That is, given $\boldsymbol{s}$, we look for a velocity vector $\boldsymbol{v'}$ such that $\boldsymbol{s}+t\boldsymbol{v'}$ is tangent to the infinite cylinder. This velocity vector is determined only up to a scaling factor, i.e., for every solution $\boldsymbol{v'}$, vector $\lambda\boldsymbol{v'}$, where $\lambda > 0$, is also a solution. We solve the scaling factor using (1). Thus we get $\boldsymbol{v'}$.

  Likewise for the recovery step, we construct the tangent from the target point $\boldsymbol{s''}$ to the infinite cylinder. That is, given $\boldsymbol{s''}$, we look for a velocity vector, $\boldsymbol{v''}$, up to a scaling factor, such that $\boldsymbol{s''}+(t''-t)\boldsymbol{v''}$ is tangent to the infinite cylinder. This yields a system of equations for the time of switch t' and the scaling factor. We solve it to get t' and $\boldsymbol{v''}$.

  We check for sanity that $0 < \tau' < t' < \tau'' < t''$, where τ' and τ'' are the times when the infinite cylinder is touched during the escape and recovery course, respectively.

- **Line/Circle.** The situation is depicted in Figure 6. The line case of the escape step is solved exactly as in the line/line case. Thus we get $\boldsymbol{v'}$.

  For the circle case of the recovery step, we determine the time θ'' of arrival at relative altitude $sign(v_z)H$.[4] For sanity, we check that $0 < \tau' < \theta'' < t''$.

  The horizontal distance to the origin at time θ'' is D. This yields an equation in t', v_x'', and v_y''. The unknowns v_x'', and v_y'' can be expressed by t', using the equations defining the time of switch. We derive and solve the quadratic equation in t', and finally get $\boldsymbol{v''}$.

- **Circle/Line**. For the circle case of the escape step, we get the time θ' of arrival at altitude $-sign(v_z)H$. We check for sanity that $0 < \theta' < t''$.

  Since at time θ' the escape course touches the boundary of the infinite cylinder, we get a system of equations in v_x' and v_y'. Using (1), we determine solutions for v_x' and v_y'. Unknowns t' and $\boldsymbol{v''}$ are then solved as in the line/line case.

- **Circle/Circle.** Circle/circle solutions (Figure 8) may exist if the times $\theta' < \theta''$ satisfy $0 < \theta'$ and $\theta'' < t''$. We get $\boldsymbol{v'}$ as in the circle/line case. Then we get t' and $\boldsymbol{v''}$ as in the line/circle case.

  We check for sanity that $\theta' < t' < \theta''$ and that the times θ' and θ'' are exit and entry points, respectively, to the protected zone.

- **Escape-Circle.** Escape-circle solutions (Figure 9) may exist if there is only one

[4] $sign(a)$ is the function that returns 1 if a is positive, -1 if a is negative, and 0 if a is zero.

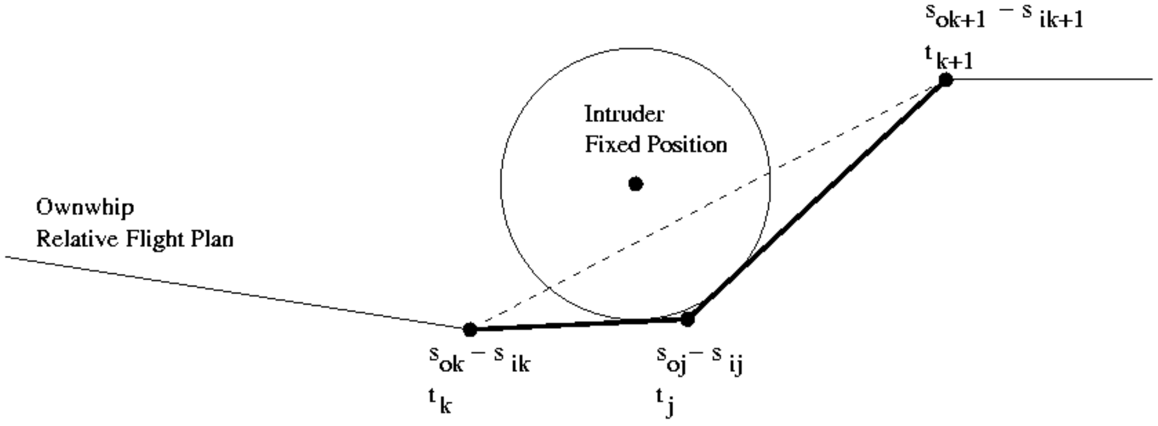

Figure 11: Strategic Conflict Detection and Resolution

intersection with a circle at time θ". We check for sanity that $0 < \theta'' < t''$.

The escape velocity vector ***v'*** is derived as in the circle/line case, but for θ" instead of θ'.

The time of switch *t'* is given by $t' = \theta''$.

We check for sanity that at time θ" there is an entry point to the infinite cylinder.

- **Recovery-Circle.** Recovery-circle solutions (Figure 10) may exist if there is only one intersection with a circle at time θ'. We check for sanity of θ' that $0 < \theta' < t''$.

  We derive solutions for ***v'*** exactly as in the circle/line case. We check for sanity that at time θ' there is an exit point from the infinite cylinder. The time of switch is given by $t' = \theta'$.

Strategic CD&R

In this section, we describe a *strategic* CD&R algorithm that uses RR3D to construct conflict-free flight plans.

We define a *flight plan* as a sequence of pairs $(\boldsymbol{s}_0,t_0)$ ... $(\boldsymbol{s}_n,t_n)$, where $t_k < t_{k+1}$, for $0 \le k < n$. The point $\boldsymbol{s}_k$ is the 3-D coordinate of the *k*-th trajectory change point, and t_k is the time when the aircraft is required to be at that point. The *segment k*, where $0 \le k < n$, is given by the subsequent points $(\boldsymbol{s}_k,t_k)$ and $(\boldsymbol{s}_{k+1},t_{k+1})$. We assume that segments describe linear trajectories, i.e., velocity vectors are constant between trajectory change points. We say that two segments in two flight plans, respectively, *correspond* if they have the same segment number.

The inputs of the strategic CD&R algorithm are the flight plans of the ownship and the intruder. Notice that we cannot apply directly our resolution and recovery algorithm (RR3D) to corresponding segments of the flight plans since they do not necessarily coincide in time. Hence, we first synchronize both flight plans in such a way that they have exactly the same number of segments and that corresponding segments coincide in time. This may be achieved by adding *virtual trajectory change points* to both flight plans. The synchronization step is performed in linear time with respect to the number of waypoints in the flight plans.

Let $(\boldsymbol{s}_{o0},t_0)$ ... $(\boldsymbol{s}_{on},t_n)$ and $(\boldsymbol{s}_{i0},t_0)$ ... $(\boldsymbol{s}_{in},t_n)$ be the synchronized flight plans of the ownship and intruder aircraft, respectively. We may assume that trajectory change points $\boldsymbol{s}_{ok}$ and $\boldsymbol{s}_{ik}$, for $0 \le k \le n$, are separated either horizontally or vertically. If this is not the case, we can always move $\boldsymbol{s}_{ok}$ out of the protected zone surrounding $\boldsymbol{s}_{ik}$.

A conflict-free flight plan for the ownship is constructed as follows. Assume that there is a conflict during the corresponding segments *k* of the ownship and intruder flight plans. We use RR3D with inputs

- $\boldsymbol{s} = \boldsymbol{s}_{ok} - \boldsymbol{s}_{ik}$
-
-
- $t'' = t_{k+1} - t_k$

to get a set of escape-recovery maneuvers. We pick one of the proposed maneuvers[5], say $(t',\boldsymbol{v}_o',\boldsymbol{v}_o'')$. This maneuver determines a new trajectory change

[5] For correctness purposes any maneuver can be selected, since RR3D produces only correct maneuvers.

point for the ownship ($\boldsymbol{s_{oj}},t_j$):

- $\boldsymbol{s_{oj}} = \boldsymbol{s_{ok}} + t'\boldsymbol{v_o}'$
- $t_j = t_k + t'$

The pair ($\boldsymbol{s_{oj}},t_j$) is inserted between the trajectory change points ($\boldsymbol{s_{ok}},t_k$), ($\boldsymbol{s_{ok+1}},t_{k+1}$) of the ownship's synchronized flight plan. Note that indeed $t_k < t_j < t_{k+1}$.

Figure 11 illustrates the situation. The dashed line is the original segment of the ownship flight plan that is in conflict with the intruder's flight plan, and the bold lines indicate the new escape and recovery segments.

Since the two flight plans are synchronized, we can formally prove that if all the corresponding segments are conflict-free then the two flight plans are also conflict-free. That is, the correctness of RR3D implies the correctness of the *strategic* conflict detection and resolution approach.

The algorithm can be summarized as follows:

- Synchronize flight plans.
- Move trajectory change points that are in conflict.
- Call RR3D for each one of the segments that are in conflict. Each time pick one solution of the choice provided by RR3D.
- Remove virtual points unless that they belong to an escape or recovery course.

In the worst case we have introduced 3 new trajectory change points for each corresponding segments in conflict: 2 virtual points due to synchronization and one trajectory change point due to the escape -recovery maneuver.

Conclusion and Future Work

We presented a *conflict resolution and recovery* algorithm, RR3D, for two aircraft in 3-D airspace. Given the position and velocity vectors of the two aircraft, ownship and intruder, and the required time of arrival of the ownship at its next target point, RR3D proposes a choice of maneuvers to the ownship. Each maneuver consists of an escape course and a subsequent recovery course that together replace the original course. The ownship is expected to perform one of the proposed maneuvers while the intruder is expected to keep its original velocity vector.

A maneuver is called *correct* if, given some reasonable assumptions about the situation, the ownship, executing the maneuver, will arrive at the target point at the scheduled time, while maintaining separation to the intruder during the whole maneuver. We gave a rigorous mathematical description of the problem and proved that RR3D proposes only correct maneuvers. We intend to check the correctness proof using the PVS theorem prover [17].

The problem may have infinitely many solutions. We restrict the solutions to finitely many by requiring that both escape course and recovery course are tangential to the intruder's protected zone and by imposing some additional constraints. In this paper we showed in some detail the constraint "change of heading", i.e., for the escape step, only the heading may change but not horizontal or vertical speed, and for the recovery step, only heading and horizontal speed may change. Other constraints and more technical details are discussed in [16].

We outlined how RR3D may be embedded in a strategic CD&R algorithm, which provides a conflict-free flight plan for the ownship. RR3D is called once per corresponding segments of the synchronized flight plan that are in conflict, and each proposed maneuver may be used to change the ownship's flight plan. The formally proven correctness of RR3D extends easily to the strategic algorithm.

A prototype of RR3D, written in Java, is currently under development. Future work includes the implementation of the strategic approach and the integration in a trajectory planner that takes into account conflicts with multiple aircraft in the same segment.

Acknowledgments. The idea to extend KB3D with recovery courses is due to David Wing. We acknowledge stimulating discussions with the DAG-TM team, Hanne Gottliebsen, Victor Carreño, Gilles Dowek, Florent Kirchner, and Ricky Butler.

References

[1] Kuchar J., L. Yang, 1997, Survey of conflict detection and resolution modeling methods, AIAA Guidance, Navigation, and Control Conference, Vol. AIAA-97-3732, New Orleans, LA, pp. 1388-1397.

[2] Radio Technical Commission for Aeronautics, 1995, Final report of the RTCA board of directors' select committee on free flight, Technical Report Issued 1-18-95, RTCA, Washington, DC.

[3] Advanced Air Transportation Technologies (AATT) Project, 1999, Concept definition for distributed air/ground traffic management (DAG-TM), version 1.0. NASA Ames Research Center - NASA Langley Research Center.

[4] Durand, N., J. -M. Alliot, J. Noailles, 1996, Automatic aircraft conflict resolution using genetic algorithms, Symposium on Applied Computing, Philadelphia, PA.

[5] Granger, G., N. Durand, J. -M. Alliot, 2001, Optimal resolution of en route conflicts, 4th USA/Europe Air Traffic Management R&D Seminar (ATM 2001), Santa Fe, NM.

[6] McDonald, J., R. Vivona, 2000, Strategic airborne conflict detection of air traffic and area hazards, Technical Report NASA Contract: NAS2-98005 RTO-29, TITAN Systems Corporation, SRC Division.

[7] Durand, N., J. -M. Alliot, F. Medioni, 2000, Neural nets trained by genetic algorithms for collision avoidance, Applied Artificial Intelligence 3 (2000).

[8] Tomlin, C., G. Pappas, S. Sastry, 1998, Conflict resolution for air traffic management: A study in multi-agent hybrid systems, IEEE Transactions on Automatic Control.

[9] Chiang, Y. -J, J. Klosowsky, C. Lee, J. Mitchell, 1997, Geometric algorithms for conflict detection/resolution in air traffic management, 36th IEEE Conference on Decision and Control.

[10] Frazzoli, E., Z. -H Mao, J. -H . Oh, E. Feron, 2001, Resolution of conflicts involving many aircraft via semidefinite programming, Journal of Guidance, Control, and Dynamics, 24(2001), pp. 79-86.

[11] Eby, M., 1994, A self-organizational approach for resolving air traffic conflicts, Lincoln Laboratory Journal, 7 (1994), pp. 239-254.

[12] Hoekstra, J., et al, 2000, Overview of NLR free flight project 1997-1999, Tech. Report NLR-CR-2000-227, National Aerospace Laboratory (NLR), The Netherlands.

[13] Bilimoria, K., 2000, A geometric optimization approach to aircraft conflict resolution, Guidance, Navigation, and Control Conference, vol. AIAA 2000-4265, Denver, CO.

[14] Dowek, G., C. Muñoz, A. Geser, 2001, Tactical conflict detection and resolution in 3-D airspace, 4th USA/Europe Air Traffic Management R&D Seminar (ATM 2001), Santa Fe, NM.

[15] Wing, D., R. Adams, B. Barmore, D. Moses, 2001, Airborne use of traffic intent information in a distributed air-ground traffic management concept: Experiment design and preliminary results, 4th USA/Europe Air Traffic Management R&D Seminar (ATM 2001), Santa Fe, NM.

[16] Geser, A., C. Muñoz, G. Dowek, F. Kirchner, 2002, Air Traffic Conflict Resolution and Recovery, NASA /CR-2002-12.

[17] Owre, S., J. Rushby, N. Shankar, 1992, PVS: A prototype verification system, 11th International Conference on Automated Deduction (CADE), D. Kapur, ed., vol. 607 of Lecture Notes in Artificial Intelligence, Saratoga, NY, Springer-Verlag, pp. 748–752.

About the Authors

Dr. Alfons Geser is interested in inductive theorem proving, term rewriting, and model checking, aheir practical application. He worked as a research assistant at the Universities of Ulm, Passau, and Tuebingen, and as an ASIC Design Engineer at a hardware design company in Passau, Germany. He joined ICASE as a Senior Staff Scientist in Jan 2001. Currently he is involved in several NASA Langley formal verification projects.

Dr. César Muñoz received his Ph.D. in Computer Science from the University of Paris 7 in 1997. After completing his Ph.D., he spent one and a half years as an International Fellow in the Formal Methods Group of the Computer Science Laboratory at SRI International in Menlo Park. He joined ICASE - NASA Langley as a Staff Scientist in May 1999. He is leading the formal verification effort on Air Traffic Management systems.