Formal verification of ACAS X, an industrial airborne collision avoidance system

Jean-Baptiste Jeannin, Khalil Ghorbal, Yanni Kouskoulas, Ryan Gardner, Aurora Schmidt, Erik Zawadzki, André Platzer

Formal Verification of ACAS X, an Industrial Airborne Collision Avoidance System

Jean-Baptiste Jeannin
Carnegie Mellon University
jeannin@cs.cmu.edu

Khalil Ghorbal
Carnegie Mellon University
kghorbal@cs.cmu.edu

Yanni Kouskoulas
The Johns Hopkins University
Applied Physics Laboratory
yanni.kouskoulas@jhuapl.edu

Ryan Gardner
The Johns Hopkins University
Applied Physics Laboratory
ryan.gardner@jhuapl.edu

Aurora Schmidt
The Johns Hopkins University
Applied Physics Laboratory
aurora.schmidt@jhuapl.edu

Erik Zawadzki
Carnegie Mellon University
epz@cs.cmu.edu

André Platzer
Carnegie Mellon University
aplatzer@cs.cmu.edu

ABSTRACT

Formal verification of industrial systems is very challenging, due to reasons ranging from scalability issues to communication difficulties with engineering-focused teams. More importantly, industrial systems are rarely designed for verification, but rather for operational needs. In this paper we present an overview of our experience using hybrid systems theorem proving to formally verify ACAS X, an airborne collision avoidance system for airliners scheduled to be operational around 2020. The methods and proof techniques presented here are an overview of the work already presented in [8], while the evaluation of ACAS X has been significantly expanded and updated to the most recent version of the system, run 13. The effort presented in this paper is an integral part of the ACAS X development and was performed in tight collaboration with the ACAS X development team.

1. INTRODUCTION

In recent years, aircraft collision avoidance has been a case study of choice for formal verification of cyber-physical systems, in particular thanks to its very clear specification – absence of collision [2, 4, 5, 12, 16, 18]. However few of those studies were done on industrial systems – i.e., systems which will ultimately equip commercial aircraft – or as an integral part of the system's development, with potential impact on the system's final design.

On the industrial side, two main systems exist, both developed under the lead of the Federal Aviation Administration (FAA): the *Traffic Collision Avoidance System* (TCAS) [3], which is currently mandated on all large passenger aircraft and whose design started in the late 1970s; and the *Next-Generation Airborne Collision Avoidance System* (ACAS X) [3, 6, 11], a complete redesign of the TCAS system initiated by the FAA to accommodate recent challenges such as more crowded airspace and the arrival of Unmanned Aerial Vehicles. This paper describes our work using hybrid systems modeling and theorem proving to formally assess the safety of ACAS X. This work is part of ACAS X's development and has been done in collaboration with the ACAS X development team. This paper focuses on what is called run 13 of the $\mathrm{X_A}$ version of ACAS X (simply denoted ACAS X throughout this paper) – the most recent version used for normal operations of manned aircraft.

A typical collision avoidance encounter involves two aircraft: the equipped *ownship* aircraft actively trying to avoid colliding with a non-equipped *intruder* aircraft. Both TCAS and ACAS X avoid such collisions by giving exclusively vertical *advisories* to the pilot of the ownship – such as "Climb at a rate of $1{,}500$ ft/min" (CL1500) or "Do not descend" (DND). If both aircraft are equipped with a collision avoidance system, both pilots can get coordinated advisories. A collision – or *Near Mid-Air Collision* (NMAC) – is formally defined as the two aircraft being within $r_p = 500$ ft horizontally and $h_p = 100$ ft vertically – thus forming a *safety puck* around the ownship where the intruder should not enter to guarantee the safety of both aircraft. While the design of TCAS was based on geometric considerations on paths taken by aircraft, the design of ACAS X is radically different. ACAS X is designed around a Markov Decision Process (MDP) estimating the probabilities that two aircraft may climb, descend or turn, in the presence or absence of an advisory [9]. Based on those probabilities, the MDP is optimized to minimize the probability of an NMAC, while also minimizing (with lesser priority) advisories, disruptive to the pilot. This optimization results in a *score table* giving advisories for different possible configurations. The table is then queried and interpolated in flight; the result is sometimes corrected with *online scores* to take into account factors not considered by the MDP such as proximity to the ground or coordination with other ACAS X-equipped aircraft, before being communicated to the pilot. The score table combines two sub-tables interpolated in succession, one with $698{,}819$ interpolation points and the other with $38{,}053{,}125$ interpolation points, resulting in a large number of combinatorial cases.

Given this design, what does it mean to formally and effectively verify ACAS X? Observe that absence of NMAC cannot be guaran-

teed in all cases: if the two aircraft are flying head-on at the same altitude and are too close to maneuver, no flyable advisory can avoid an NMAC. Moreover, directly verifying each point in the ACAS X score table is infeasible because of the size of the table. Instead, we seek to formally quantify the behavior of ACAS X. Our approach focuses on the score table and is composed of three major steps. First, based on a model of the typical behavior of the aircraft, we symbolically characterize *safe regions*, geometric configurations of the ownship and intruder's positions and speeds such that a given advisory, if followed correctly, will not result in an NMAC. Second, we model both aircraft as a hybrid system combining differential equations representing *continuous* motion of the aircraft, and *discrete* actions modeling advisories – issued according to the safe zones – and compliance of the pilot. Using this model, we *formally prove* that, if the two aircraft are in a safe region, following the advisory associated with that safe region will not result in an NMAC. Third and finally, we exhaustively *compare* the safe regions to the advisories given by the score table, thus transferring our formal argument of safety to the operational ACAS X. We identify parts of the table that are provably safe, but also counterexamples of situations where following the table could lead to an NMAC. We feed back those results to the ACAS X development team. The general formal verification approach used in this work – modeling, proving and comparing – is not limited to ACAS X, but can potentially be reused for other collision avoidance systems, or even other systems whose design is based on optimization or machine learning.

Related Work. Our approach is different from previous complementary work in various ways. We analyze an independent industrial system instead of hypothetical safe-by-design systems as in [2, 4, 5, 12, 16, 18]. Our hybrid model uses realistic, continuous dynamics, and does not discretize the trajectories of aircraft as in [2, 19, 4]. We provide mechanized proofs of correctness of our model instead of simulations or stress testing approaches as in [1, 6, 10, 13, 18]. Finally, unlike [19, 9], our hybrid model and proofs are independent from the different versions of the score table used by ACAS X, thus allowing us to provide universal safe regions that can be reused for ACAS X or even new systems. A more detailed discussion of related work can be found in [8].

This paper gives a brief overview of the modeling approach in Sect. 2 and focuses on the comparison and its results in Sect. 3. Our ongoing work and future avenues are reported in Sect. 4.

2. OVERVIEW OF THE APPROACH

We present and motivate the hybrid model and construction of the safe regions. We sketch the main steps of our formal proof of safety. For more details, we refer the reader to [8] and its companion technical report [7].

2.1 Modelling ACAS X

Let us consider a 3-dimensional encounter involving an ownship O and an intruder I. For the purposes of this study, we make a number of assumptions on the capabilities and behavior of both aircraft during an encounter. We assume that the ownship has the capability to *perfectly* estimate the geometrical configuration of the encounter, i.e., the relative position of the intruder, as well as the speed and direction of both aircraft; in practice such information is provided by different sensors as well as communication data exchanged between aircraft. In particular, we discard all sources of noise and uncertainty related to the state estimation. We further assume that the intruder has a constant velocity vector, both horizontal and vertical, with no turn, acceleration or change in its vertical speed, thus describing a *straight-line uniform* trajectory. As for the ownship, we assume that it has a *straight-line uniform horizontal* trajectory, with no turn or horizontal acceleration. We, however, allow non-deterministic changes in the ownship vertical acceleration that are used to come into compliance with and follow vertical advisories. These assumptions limit the generality of the result, but the proof presented in this paper is still powerful because most planes, most of the time, fly in straight lines; therefore we expect ACAS X to be able to handle this sort of steady-state pilot model. We make these assumptions throughout the technical development of the paper, including in Sect. 3.

ACAS X can issue one of 16 vertical advisories, summarized in Table 1. For example, if provided with the (corrective) advisory "Climb 1,500" (CL1500), the pilot is instructed to climb at a rate of at least $1,500$ ft/min. Similarly, the (preventive) advisory "Do Not Climb" (DNC) instructs the pilot to not climb, while the advisory "Maintain Climb" (MCL) instructs the pilot to maintain their current climb rate. Our model, based on [9], supposes that the pilot complies with the advisory with acceleration at least $a_r = g/4$ starting after a delay modeling reaction time of at most $d_p = 5$ seconds; then the pilot continues following the advisory forever – or until a new advisory is issued. Four advisories (SDES1500, SCL1500, SDES2500 and SCL2500) can only be used to strengthen a previous advisory; as such the compliance of the pilot is typically better, and our model supposes a compliance with an acceleration of $a_r = g/3$ and within a delay of $d_p = 3$ seconds. ACAS X can also issue the advisory "Clear of Conflict" (COC), meaning that no specific action is required from the pilot. All advisories, except COC, request the pilot to stay either above or below a given vertical velocity, as shown in Table 1. We model such advisories with two variables: the target vertical velocity given to the pilot $\dot{h}_f$ and a binary variable w, taking value -1 if the pilot is asked to stay below that velocity, or $+1$ if she is asked to stay above.

2.2 Reduction to a 2-dimensional Encounter

Under our assumptions, the 3-dimensional encounter can be reduced to a 2-dimensional encounter without loss of generality. This observation makes the formal model much easier to reason about. This section gives a geometric intuition of the reduction. A formal proof, based on a similar argument, is presented in [8].

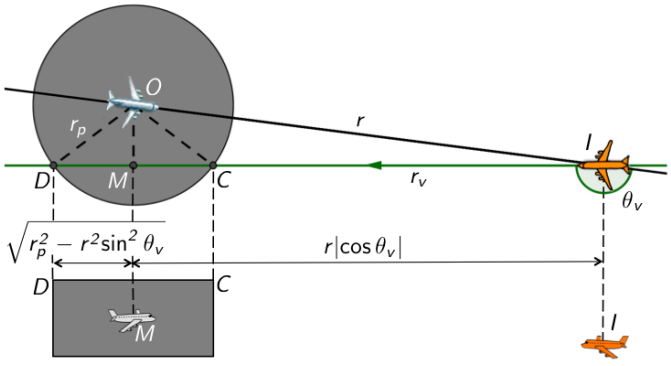


Figure 1: Reduction to a 2-dimensional encounter

For this reduction let us put ourselves in the reference frame of the ownship. A 3-dimensional encounter is represented in Fig. 1 – top view at the top and side view at the bottom. Let r be the distance between the aircraft, $\vec{r_v}$ the relative velocity of the intruder with respect to the ownship, and θ_v the angle formed by (OI) and $\vec{r_v}$.

Table 1: Advisories and their modeling variables [8, 7]

	ACAS X Run 13 Specification [9]				Our model	
	Vertical Rate Range		Strength	Delay (s)		
Advisory	Min (ft/min)	Max (ft/min)	a_r	d_p	w	$\dot{h}_f$ (ft/min)
DNC2000	$-\infty$	$+2000$	$g/4$	5	-1	$+2000$
DND2000	-2000	$+\infty$	$g/4$	5	$+1$	-2000
DNC1000	$-\infty$	$+1000$	$g/4$	5	-1	$+1000$
DND1000	-1000	$+\infty$	$g/4$	5	$+1$	-1000
DNC500	$-\infty$	$+500$	$g/4$	5	-1	$+500$
DND500	-500	$+\infty$	$g/4$	5	$+1$	-500
DNC	$-\infty$	0	$g/4$	5	-1	0
DND	0	$+\infty$	$g/4$	5	$+1$	0
MDES	$-\infty$	current	$g/4$	5	-1	current
MCL	current	$+\infty$	$g/4$	5	$+1$	current
DES1500	$-\infty$	-1500	$g/4$	5	-1	-1500
CL1500	$+1500$	$+\infty$	$g/4$	5	$+1$	$+1500$
SDES1500	$-\infty$	-1500	$g/3$	3	-1	-1500
SCL1500	$+1500$	$+\infty$	$g/3$	3	$+1$	$+1500$
SDES2500	$-\infty$	-2500	$g/3$	3	-1	-2500
SCL2500	$+2500$	$+\infty$	$g/3$	3	$+1$	$+2500$
COC	$-\infty$	$+\infty$	Not applicable			

The green line (DC), along the vector $\vec{r_v}$ represents the horizontal trajectory of the intruder I relative to the ownship O. The gray disk surrounding the ownship represents the projected safety puck. Recall that to avoid an NMAC, the trajectory of the intruder must not intersect the safety puck. If $r|\sin\theta_v| > r_p$, then the green projected trajectory of the intruder does not intersect the projected safety puck and there is no possible NMAC. Otherwise, let us look at a side view of the vertical plane along the trajectory (DC), represented at the bottom of Fig. 1. In this plane, the intruder avoids an NMAC if and only if it does not enter a *virtual puck* centered on a *virtual ownship* M middle of $[DC]$, with a modified puck radius $\sqrt{(r^2 - r_p^2\sin^2\theta_v)}$ and a modified distance to M, $r|\cos\theta_v|$. This reduces the 3D encounter to a virtual planar head-on encounter.

2.3 Construction of the Safe Regions

Let us now change the reference frame to put ourselves in the reference frame fixed to the intruder and centered on the original position of the ownship. Let h be the relative height of the intruder with respect to the ownship, $\dot{h}_0$ the relative vertical velocity of the ownship with respect to the intruder, and a the relative vertical acceleration of the ownship with respect to the intruder.

Let us consider an ownship initially descending and receiving a CL1500 advisory, meaning that the pilot should start climbing at 1,500 ft/min. Let us further assume that the pilot reacts immediately. The nominal trajectory of the ownship is illustrated in red on Fig. 2: it consists of a parabola at acceleration $g/4$ until reaching the advised 1,500 ft/min, followed by a straight line climbing at 1,500 ft/min. If the pilot follows the advisory, the ownship will always be at or above this nominal trajectory. This nominal trajectory gives rise to a first construction of what we call the *safe region*: if the intruder is in the green area to the right, left or below all pucks centered on a point of the red trajectory, then the intruder is in a safe region, and no NMAC can occur as long as the straight line encounter assumptions are satisfied. Based on the nominal trajectory, we can mathematically express a symbolic *implicit* formulation of the safe region in full generality, given in Fig. 3.

The implicit formulation involves three universal quantifiers over t, r_t and h_t. Eliminating those quantifiers is crucial when it comes to comparing the actual advisories of ACAS X. Quantifier elimination algorithms could be used but often lead to formulations that are very large and hard to interpret. Instead, we go back to Fig. 2 and manually construct the boundaries of the safe region. They consist of a vertical line on the left side created by the left side of the puck, followed by a half parabola created by the bottom left-hand corner of the puck, then a horizontal segment of length $2r_p$ created by the bottom of the puck, and finally another half parabola and a straight-line at 1,500 ft/min, both created by the bottom right-hand corner of the puck. This observation leads to an *explicit* formulation of the safe regions which can be expressed mathematically in full generality and is given in Fig. 3.

Both formulations are equivalent, but each comes with an advantage: the implicit formulation makes the formal proof of the hybrid model easier, while the explicit formulation enables an easy comparison with the ACAS X table. We formally prove in the KeYmaera theorem prover [17] that the implicit formulation of the safe region $C_{\mathsf{impl}}(r,h,\dot{h}_0)$ and its explicit formulation $C_{\mathsf{expl}}(r,h,\dot{h}_0)$ are equivalent. This formal proof, along with all the KeYmaera models and proofs of this paper are available online at `http://www.ls.cs.cmu.edu/pub/acasx.zip`.

2.4 Proof of Safety

In order to formally prove the correctness of the safe regions, we build in Eq. (1) a hybrid model described by the differential dynamic logic [15]. We then formally prove, using the hybrid systems theorem prover KeYmaera [17], that no NMAC can occur, provided all advisories satisfy the condition $C_{\mathsf{impl}}(r,h,\dot{h}_0)$ (or equivalently $C_{\mathsf{expl}}(r,h,\dot{h}_0)$).

$$
\begin{array}{l}
r_p \geq 0 \wedge h_p > 0 \wedge r_v \geq 0 \wedge a_r > 0 \wedge \\
(w = -1 \vee w = 1) \wedge C_{\mathsf{impl}}(r,h,\dot{h}_0) \rightarrow \\
[(\ (\ ?\mathsf{true} \cup \\
\quad \dot{h}_f := *; (w := -1 \cup w := 1); \\
\quad ?C_{\mathsf{impl}}(r,h,\dot{h}_0); \mathsf{advisory} := (w, \dot{h}_f)\ ); \\
\quad a := *; \\
\quad \{r' = -r_v, h' = -\dot{h}_0, \dot{h}_0' = a \ \&\ w\dot{h}_0 \geq w\dot{h}_f \vee wa \geq a_r\} \\
)^*]\ (|r| > r_p \vee |h| > h_p)
\end{array}
\tag{1}
$$

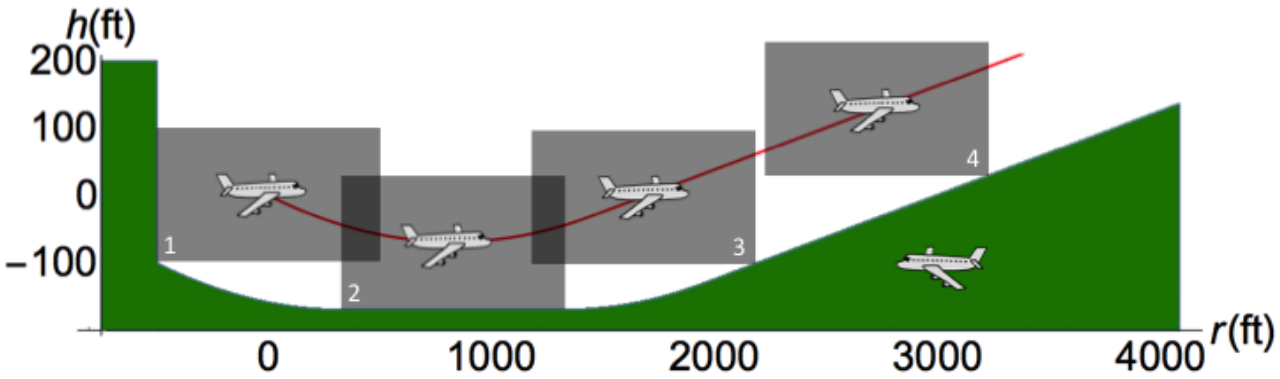


Figure 2: Trajectory of ownship (red) and safe region for the intruder (green), immediate response [8]

Implicit formulation

$$A(t,h_t,\dot{h}_0) \equiv \left(\begin{array}{rcl} 0 \le t < \dfrac{\max(0,w(\dot{h}_f-\dot{h}_0))}{a_r} & \wedge & h_t = \dfrac{wa_r}{2}t^2+\dot{h}_0 t \end{array}\right) \vee \left(\begin{array}{rcl} t \ge \dfrac{\max(0,w(\dot{h}_f-\dot{h}_0))}{a_r} & \wedge & h_t = \dot{h}_f t - \dfrac{w\max(0,w(\dot{h}_f-\dot{h}_0))^2}{2a_r} \end{array}\right)$$

$$C_{\mathsf{impl}}(r,h,\dot{h}_0) \equiv \forall t.\forall r_t.\forall h_t.\Big(r_t = r_v t \wedge A(t,h_t,\dot{h}_0) \rightarrow (|r-r_t| > r_p \vee w(h-h_t) < -h_p)\Big)$$

Explicit formulation

$$\mathsf{case}_1(r,\dot{h}_0) \equiv -r_p \le r < -r_p - \frac{r_v \min(0,w\dot{h}_0)}{a_r} \qquad \mathsf{case}_2(r,\dot{h}_0) \equiv -r_p - \frac{r_v \min(0,w\dot{h}_0)}{a_r} \le r \le r_p - \frac{r_v \min(0,w\dot{h}_0)}{a_r}$$

$$\mathsf{bound}_1(r,h,\dot{h}_0) \equiv wr_v^2 h < \frac{a_r}{2}(r+r_p)^2 + wr_v\dot{h}_0(r+r_p) - r_v^2 h_p \qquad \mathsf{bound}_2(r,h,\dot{h}_0) \equiv wh < -\frac{\min(0,w\dot{h}_0)^2}{2a_r} - h_p$$

$$\mathsf{case}_3(r,\dot{h}_0) \equiv r_p - \frac{r_v \min(0,w\dot{h}_0)}{a_r} < r \le r_p + \frac{r_v \max(0,w(\dot{h}_f-\dot{h}_0))}{a_r} \qquad \mathsf{case}_4(r,\dot{h}_0) \equiv r_p + \frac{r_v \max(0,w(\dot{h}_f-\dot{h}_0))}{a_r} < r$$

$$\mathsf{bound}_3(r,h,\dot{h}_0) \equiv wr_v^2 h < \frac{a_r}{2}(r-r_p)^2 + wr_v\dot{h}_0(r-r_p) - r_v^2 h_p$$

$$\mathsf{bound}_4(r,h,\dot{h}_0) \equiv (r_v = 0) \vee \left(wr_v h < w\dot{h}_f(r-r_p) - \frac{r_v \max(0,w(\dot{h}_f-\dot{h}_0))^2}{2a_r} - r_v h_p\right)$$

$$\mathsf{case}_5(r,\dot{h}_0) \equiv -r_p \le r < -r_p + \frac{r_v \max(0,w(\dot{h}_f-\dot{h}_0))}{a_r} \qquad \mathsf{case}_6(r,\dot{h}_0) \equiv -r_p + \frac{r_v \max(0,w(\dot{h}_f-\dot{h}_0))}{a_r} \le r$$

$$\mathsf{bound}_5(r,h,\dot{h}_0) \equiv wr_v^2 h < \frac{a_r}{2}(r+r_p)^2 + wr_v\dot{h}_0(r+r_p) - r_v^2 h_p$$

$$\mathsf{bound}_6(r,h,\dot{h}_0) \equiv (r_v = 0 \wedge r > r_p) \vee \left(wr_v h < w\dot{h}_f(r+r_p) - \frac{r_v \max(0,w(\dot{h}_f-\dot{h}_0))^2}{2a_r} - r_v h_p\right)$$

$$C_{\mathsf{expl}}(r,h,\dot{h}_0) \equiv \left(w\dot{h}_f \ge 0 \rightarrow \bigwedge_{i=1}^{4}(\mathsf{case}_i(r,\dot{h}_0) \rightarrow \mathsf{bound}_i(r,h,\dot{h}_0))\right) \wedge \left(w\dot{h}_f < 0 \rightarrow \bigwedge_{i=5}^{6}(\mathsf{case}_i(r,\dot{h}_0) \rightarrow \mathsf{bound}_i(r,h,\dot{h}_0))\right)$$

Figure 3: Implicit and explicit formulations of the safe region for an immediate response [8]

Eq. (1) is of the form $p \rightarrow [\alpha]q$, which says all executions of program α starting in a state satisfying logical formula p end up in a state satisfying q. It is akin to the Hoare triple $\{p\}\alpha\{q\}$ with precondition p and postcondition q. The precondition in Eq. (1) imposes constraints on several constants, as well as the formula $C_{\mathsf{impl}}(r,h,\dot{h}_0)$ that forces the intruder to be in a safe region for an initial advisory $(w,\dot{h}_f)$. We cannot guarantee safety if the intruder starts in an unsafe region. The postcondition encodes absence of NMAC. Lines 3 to 5 express the action of ACAS X. The nondeterministic choice operator $\cup$ expresses that the system can either continue with the same advisory by doing nothing – testing the trivial condition ?true – which ensures it always has a valid choice and cannot get stuck. Otherwise it can choose a new advisory $(w,\dot{h}_f)$ that passes the safety condition $C_{\mathsf{impl}}(r,h,\dot{h}_0)$ – the variable "advisory" will be the next message to the pilot. Lines 6 and 7 express the action of the ownship, first nondeterministically choosing an arbitrary acceleration ($a := *$) then following the continuous dynamics. The evolution of the variables r, h and $\dot{h}_0$ is expressed by a differential equation, and requires (using the operator &) that the ownship evolves towards its target vertical velocity $\dot{h}_f$ at acceleration a_r (condition $wa \ge a_r$), unless it has already reached vertical velocity $\dot{h}_f$ (condition $w\dot{h}_0 \ge w\dot{h}_f$). Finally, the star * on line 8 indicates that the program can be repeated any number of times, allowing the system to go through several safe advisories.

2.5 Delayed Pilot Response and "Clear of Conflict" Advisory

A similar process can be followed to add a delay parameter that models the reaction time of the pilot before she starts following the advisory. During that delay we consider a worst case scenario where the pilot can climb or descend at an acceleration of $g/3$. Details of the approach, including a formally proved model of the system and equations of the safe regions, can be found in [8].

This approach can be adapted to model a "Clear of Conflict" (COC) advisory. When a COC advisory is issued, our model allows the pilot to climb or descend with an acceleration up to $g/3$. A COC advisory can be considered safe if and only if it can be followed by another safe advisory 1 second later – since ACAS X can issue advisories every second. Thus, COC is safe if and only if there exists an advisory that is safe for a delay augmented by 1 second.

COC has the same representation as other advisories in the model – a period of delay followed by compliance with an advisory – but has a different interpretation. The COC advisory itself only advises the pilot that the $g/3$ action for the delay period in the model is safe. The vertical velocity limit attached to COC represents a future safe advisory that may be issued after the COC to ensure safety after the COC period is over. As such, the value of its vertical velocity limit is variable, and depends on what could be safely issued by the system after a period of pilot delay and free vertical acceleration.

2.6 Limitations

Our approach and model have some limitations. Our model of a straight-line uniform trajectory for the intruder disallows any turn or reaction on its part; in particular it does not handle an intruder equipped with a collision avoidance system. Similarly, our model does not allow any turn on the part of the ownship. As a result, our safe regions do not guarantee NMAC avoidance for different, non-straight-line pilot behavior, and not satisfying our safety conditions does not necessarily lead to an NMAC. Furthermore, our model permits but does not enforce follow-on advisories. Our analysis thus conservatively rejects advisories that are not safe for all future, even those that ACAS X would ultimately strengthen or reverse. But follow-on advisories can change the outcome of the encounter if there is enough time for the pilot to react to the modified advisory.

In addition of the advisories presented in Table 1, ACAS X can issue the recently-added Multi-Threat Level-Off (MTLO) advisory that we do not handle yet in our model or safe regions. Finally, our comparison with the ACAS X system focuses on its score table exclusively, and ignores online scores that account for example for ground proximity or coordination.

3. COMPARISON

For each configuration of the aircraft, the safety conditions presented in Fig. 3 give us a means of evaluating the safety of the ACAS X table, by performing a *comparison* between the provably safe set of advisories allowed by the safety conditions, and the advisory recommended by the ACAS X table. Any mismatch gives us a *counterexample* of a potentially unsafe behavior. This section presents the method and results of the comparison, then examines a few automatically generated counterexamples.

3.1 Method

The ACAS X score table used to determine advisories is stored as a (high-dimensional) grid of points, often called cutpoints. When the ACAS X table is queried online, it performs a multilinear interpolation to combine the table data from the surrounding cutpoints. Thus, in a sense, all of the information in the table is determined by its cutpoints. As a first step in analyzing the safety of the ACAS X table, we compare the advisory given by the system for each cutpoint with the set of advisories that are proved safe in our alternate model. Throughout this section we call a cutpoint *safe* if its corresponding ACAS X advisory respects the safe regions.

The comparison we perform covers all of the cutpoints in the table exhaustively (except for the recently-added advisory MTLO). For the run 13 of ACAS X, cutpoints are defined by: relative horizontal motion (angle and magnitude, with 187 and 37 discrete values); horizontal separation (101 discrete distances); vertical velocities (25 discrete vertical velocities each for ownship and intruder); relative vertical separation (45 discrete distances); the current state of the advisory, and whether the system estimates it is being followed (33 distinct states). Some continuous dimensions have equally spaced cutpoints (e.g. θ_v is equally spaced in radians), but others have variable spacing, with points closer in areas that may be more dangerous (e.g. vertical separation is discretized in steps of $50\,\text{ft}$ between $-400\,\text{ft}$ and $400\,\text{ft}$, but the discretization step then becomes increasingly larger – $100\,\text{ft}$, $200\,\text{ft}$, $400\,\text{ft}$, and finally $4{,}000\,\text{ft}$ at the last step – as the separation increases). All in all, there are $648{,}591{,}384{,}375$ (6.49e11) unique cutpoints to analyze. Because the table may have arbitrarily different entries at each cutpoint, we cannot rely on any sort of symmetry arguments, but have to explore the full state space exhaustively. Fortunately however, we do not have to predict all of the infinitely many possible future trajectories from a cutpoint to check whether an advisory is safe, but we can exploit the safe regions identified in Sect. 2 to reduce this to a simple arithmetic check of the formula in Fig. 3.

Improving upon our previous paper [8], we present new results and counterexamples featuring a delayed pilot response within the limits assumed in Sect. 2, and counterexamples where the system issued COC. Because the safe regions are symbolic formulas, the same comparison can be repeated with any other instantiation of the parameters such as the worst case vertical acceleration (assumed to be $g/3$) or the pilot delay (3 seconds). The proofs for the safe regions identify them as safe for any instantiation within the model, which shows a clear advantage of the formal symbolic approach whenever achievable.

When the system issues a COC, the comparison computes a two-sided envelope, since the system isn't asking the pilot to go in any particular direction. Each side of the envelope represents the most extreme acceleration in that direction during the delay ($g/3$), followed by the most extreme advisory after 1 second in the other direction (either DES1500 or CL1500). The upper and lower bounds of the envelope eventually cross to create a closed area, whose inside is considered unsafe. The area outside this curve is safe for any intruder, and guarantees we can avoid collision if we issue COC now, by issuing the appropriate follow-on advisory the next second.

3.2 Results

We ran two different analyses, one comparison assuming that the pilot reacts immediately (no delay), and another one assuming that the pilot reacts after a 3-second delay. We present the results in Table 2. For the 3-second-delay comparison, 97.7% of the cutpoints we examined were either safe, or had no alternate advisory for which they could be made safe. Only about 2.3% of the cutpoints had alternate advisories that could be issued to improve the table's

Table 2: Results of the comparison

	With 3-second delay		With no delay	
	Number of cutpoints	Percentage	Number of cutpoints	Percentage
Points examined	648,591,384,375	100.0%	648,591,384,375	100.0%
Safe or unresolvable points	633,430,949,641	97.7%	632,951,653,646	97.6%
Counterexamples	15,160,434,734	2.3%	15,639,730,729	2.4%

behavior in straight-line encounters. For the no-delay comparison, we found similar statistics of 97.6% and 2.4%, respectively.

3.3 Counterexamples

We discuss four counterexamples found by the comparison, using either no delay for pilot response, or a 3-second delay prior to compliance. The counterexamples are visualized by plotting the predicted ownship and intruder paths in altitude versus time, afer horizontal reduction if necessary (Figures 4 and 5). When visualizing the counterexample, a particular acceleration and vertical rate target is chosen during the delay period that when combined with later compliance to an advisory, results in an NMAC.

The two panels of Fig. 4 show examples where the ACAS X advisory appears to increase the risk of an NMAC over the nominal risk had the pilot continued on the same trajectory. Panel (a) shows a counterexample found using no delay in pilot response. In this case, the ownship was originally planning to cross in altitude with the intruder. The table issues a DNC advisory, causing the ownship to reduce its upward velocity to zero at a rate of $g/4$. This results in a collision course with the intruder and an NMAC approximately 20 seconds later. After inspecting the ACAS X table, it was determined that a major cause for this behavior was inaccurate horizontal timing estimation in the table. As a result of this counterexample, improvements are currently being made to the system's timing estimation components, thus demonstrating the benefit of applying formal verification techniques early on in the development.

Panel (b) of Fig. 4 shows a counterexample where the advisory issued induces a risk for a pilot that responds after a delay. In the example plotted, during the delay of 3 seconds, the ownship pilot increases climbing with acceleration $g/3$ to reach 5,000 ft/min, before complying with the DNC advisory with acceleration $g/4$. This possible behavior puts the ownship on a near collision course with the intruder. Based on the anticipated miss distance of the nominal course, advisories CL1500 or MCL would have lowered the risk here. These counterexamples where ACAS X advisories appear to induce a problem are collected and communicated to the system designers for studying and improving the system.

Fig. 5 shows counterexamples where the ACAS X table does not issue an advisory (i.e., issues COC), but the scenario would be significantly safer with an advisory for straight-line encounters. Panel (a) shows an identified counterexample where, due to the very close horizontal range of only 150 feet, the intruder is already within horizontal conflict range. However, the intruder begins 1,000 feet above the ownship and is descending. Without an advisory, the ownship could decide to climb for 1 second before an advisory can be issued by the system. Even if the most extreme available advisory (DES1500) were issued 1 second later, the intruder would enter the back edge of the puck. If, however, a descend advisory were issued in the first place and the pilot responds with no delay, then the observed NMAC be avoided.

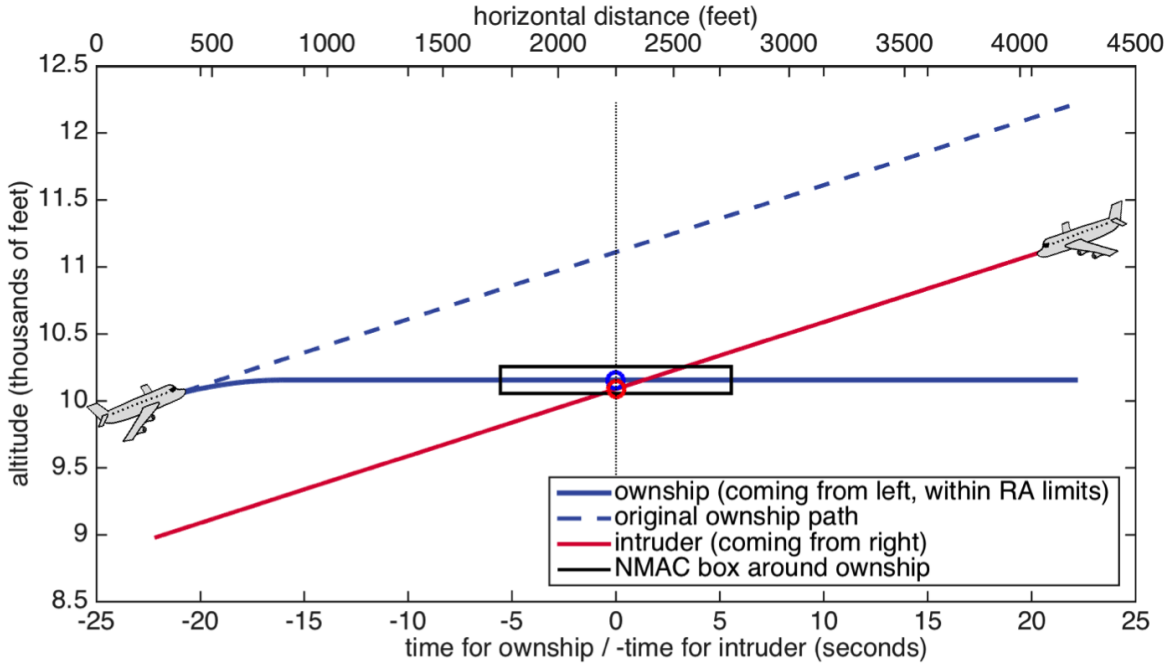
In panel (b) of Fig. 5, due to the high horizontal closure rate the time to horizontal conflict is quite short (about 7 seconds). By not issuing an advisory at this state, the pilot may reduce climb to only $8,000$ ft/min during the 1 second where no advisory is issued and keep the same vertical velocity during the assumed delay of 3 seconds after a corrective advisory, such as SDES1500 (descend at rate 1,500 ft/min) is issued. By the time the pilot starts complying with the advisory it is already too late to avoid NMAC. Due to the high vertical rates of this example, the descend advisories are the only type of advisories available to ACAS X, because the system is designed to not increase the climbing rate beyond 2,500 ft/min. These two counterexamples show identified states where not issuing an advisory may actually augment the risk of NMAC.

The comparison we are performing generates billions of counterexamples. By categorizing and using them to create simulation tests, we provide an important body of stress testing scenarios. Analysis and discussion of counterexamples with system designers has already led to design improvements of the ACAS X system.

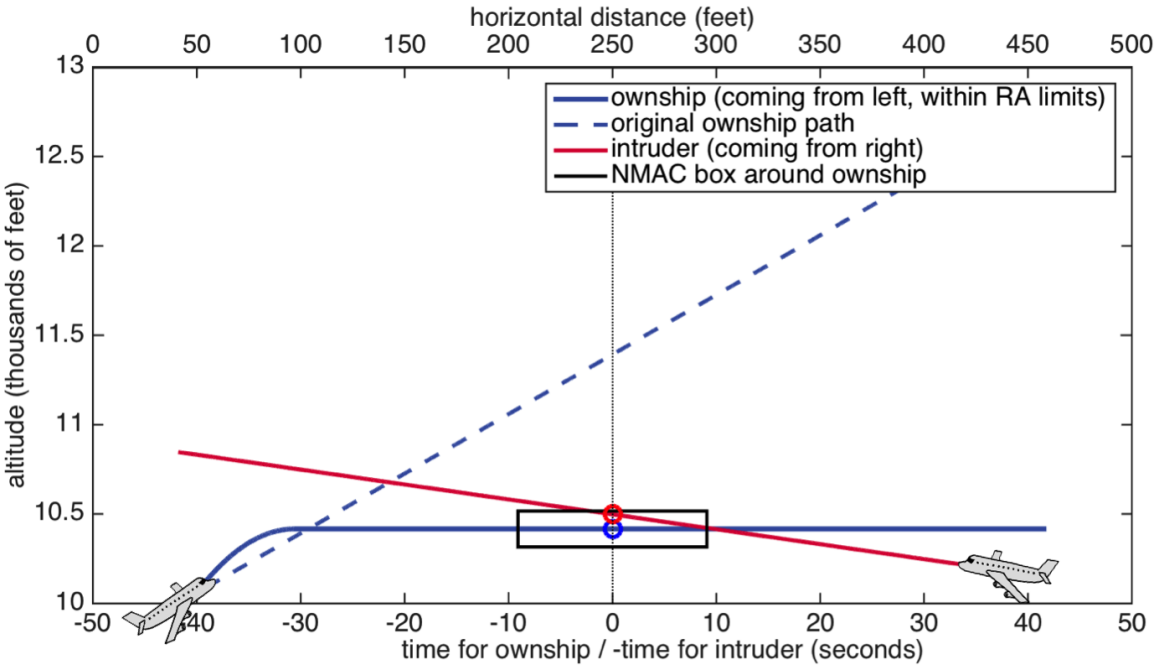
4. CURRENT AND FUTURE WORK

Aircraft collision avoidance systems such as ACAS X have a huge potential to make air travel safer and more efficient. They can help utilize the increasingly crowded airspace more efficiently. And when aircraft get too close to one another, they can help the pilot prevent a mid-air collision. Aircraft collision avoidance systems are expected to give good advisories under all circumstances while simultaneously being minimally invasive for the pilot. Among all possible advisories, the system is supposed to prefer choices that minimize the impact on the original flight trajectories while also avoiding the potential of NMAC. The advisories have to be issued fast enough so that they still resolve the NMAC before it is too late, respecting that the pilot may take a moment to react to the advisory and have flexibility in how exactly they follow it. And the system should not give advisories that could confuse the pilot under stress; for example, it should avoid changing its mind too quickly about whether to climb or descend. Furthermore, the system should work well whether the intruder is equipped with a collision avoidance system or not. These are a lot of responsibilities and safety requirements for a system like ACAS X, which make its design very challenging. How could it ever be possible to design a system respecting those constraints? And, once designed, how could one argue that it will really work as intended under all circumstances?

The safety analysis of aircraft collision avoidance systems is challenging, because of the many possible geometric configurations to consider, as well as complicated predictions of how the positions of the aircraft may change over time. Additionally, ACAS X uses a large machine-optimized score table for its decisions. This setting leads to a disciplined way of reaching optimal compromises between safety and operational suitability, but makes verification more difficult. For one thing, classical testing techniques that try to reach some kind of coverage (e.g. one test case per path through a program) are hopelessly insufficient for ACAS X, because most of its crucial decisions come from interpolating the individual data entries in the table. Testing what happens when following all paths

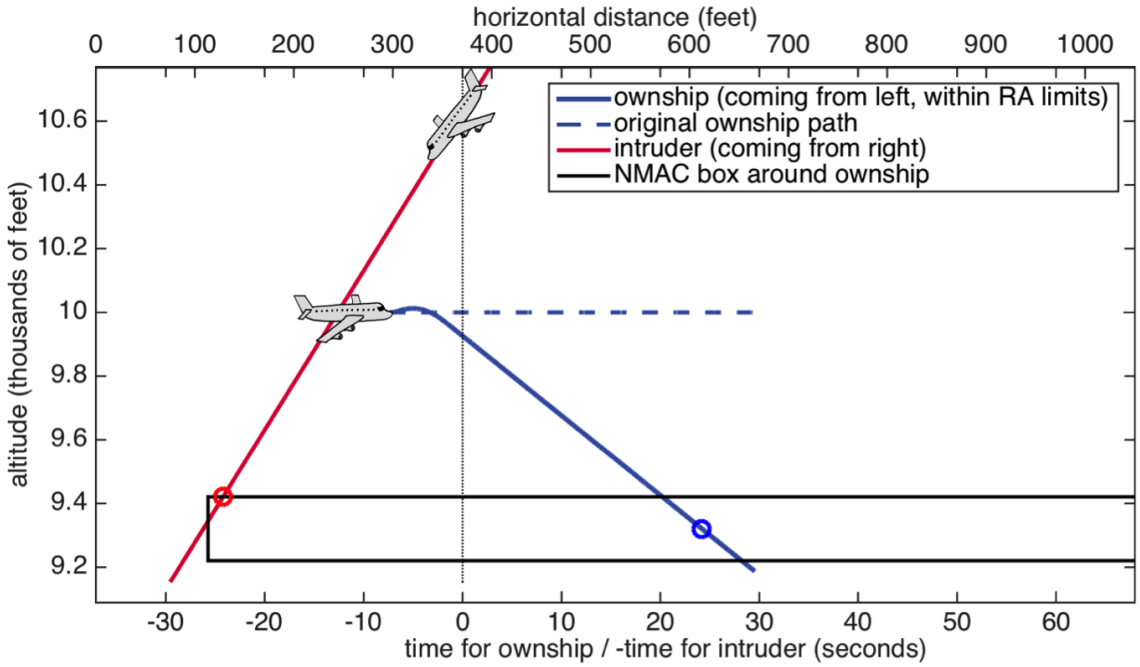


(a) Pilot responds immediately to the advisory DNC (Do Not Climb). Starting state is $r = 4{,}000$ ft, $r_v = 180$ ft/s , $\theta_v = 180°$, $h = 1{,}200$ ft, $\dot{h}_0 = 3{,}000$ ft/min, $\dot{h}_1 = -3{,}000$ ft/min, previous state = 'None-None'. There is an NMAC at time 0.00 s with 66.66 ft vertical separation and 0.40 ft horizontal separation.

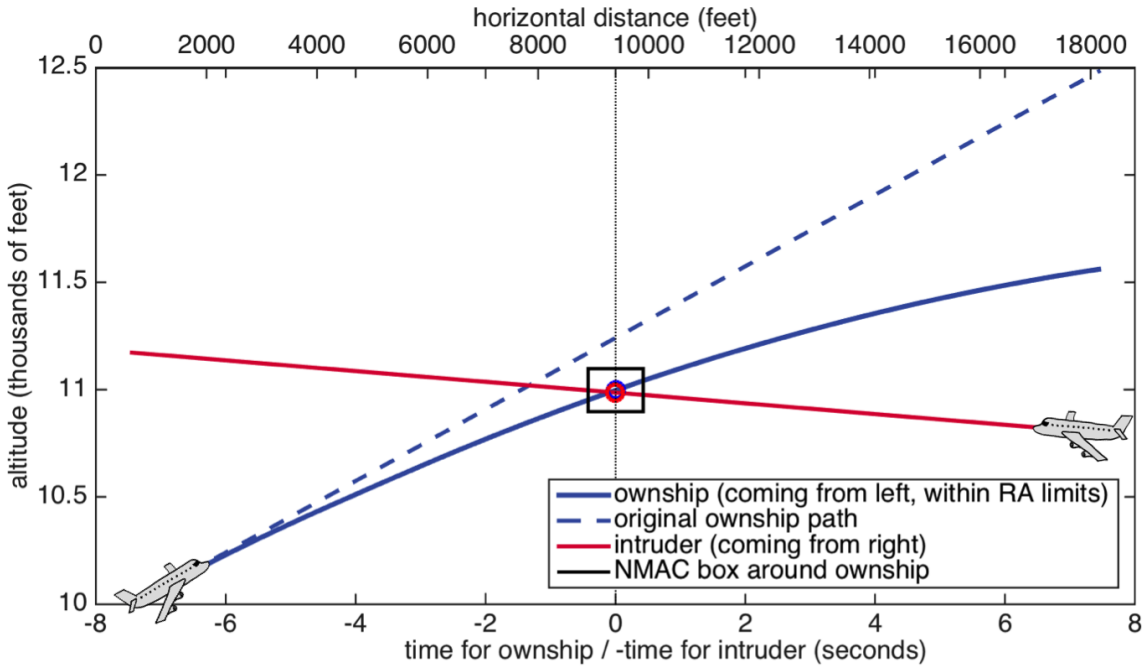


(b) Pilot responds to the advisory DNC after 3 seconds. Starting state is $r = 650$ ft, $r_v = 10$ ft/s , $\theta_v = 130°$, $h = 150$ ft, $\dot{h}_0 = 2{,}000$ ft/min, $\dot{h}_1 = 500$ ft/min, previous state = 'None-None'. There is an NMAC at time 0.00 s with 81.79 ft vertical separation and 0.01 ft horizontal separation.

Figure 4: ***Counterexamples with Induced Risk*****: Original ownship path (dashed blue) and intruder path (red) vs. potential ownship path (blue) responding to a do-not-climb (DNC) advisory with no delay (panel a) and with a 3-second delay (panel b).**



(a) Pilot responds immediately to COC (no advisory) followed 1 s later by Descend 1,500 ft/min. Starting state is $r = 150$ ft, $r_v = 20$ ft/s , $\theta_v = 170°$, $h = 1{,}000$ ft, $\dot{h}_0 = 0$ ft/min, $\dot{h}_1 = -3{,}000$ ft/min, previous state = 'None-None'. There is an NMAC at time 24.20 s with 99.84 ft vertical separation and 484 ft horizontal separation.



(b) Pilot responds to COC (no advisory) after 3 s followed 1 s later by Subsequent-Descend 1,500 ft/min. Starting state is $r = 17{,}500$ ft, $r_v = 2{,}350$ ft/s $\theta_v = 180°$, $h = 800$ ft, $\dot{h}_0 = 10{,}000$ ft/min, $\dot{h}_1 = 1{,}500$ ft/min, previous state = 'DND-DND'. There is an NMAC at time 0.00 s with 10.99 ft vertical separation and 7.50 ft horizontal separation.

Figure 5: ***Counterexamples of Insufficient Advice*****: Original ownship path (dashed blue) and intruder path (red) vs. potential ownship path (blue) responding to a COC advisory, i.e., continuing with no advisory for 1 second followed with the most extreme advisory available with no delay (panel a) and with 3-second delay (panel b).**

through the table would be impossible and not even necessarily conclusive, even if performed at approximate grid resolutions.

This paper illustrates that the verification problem for ACAS X is not completely insurmountable, however. An appropriate separation of concerns in the verification design made it possible to verify collision freedom of controls respecting safe regions, under a broad class of behaviors in a precisely specified hybrid system model, and then subsequently compare those safe regions to the decisions of the score table. This is a big step forward in the safety analysis of ACAS X and contributed to a deeper understanding of ACAS X itself. At the same time, the formal verification of ACAS X presented in this paper can be improved in several different ways.

More Accurate Model. The hybrid model presented in Sect. 2 has a number of limitations. In many of our counterexamples, ACAS X would issue an advisory later on, correcting its initial assessment. In contrast, our current model assumes that an advisory is followed forever. We are working on an improved model handling such *subsequent advisories*. Our model also assumes that both aircraft follow a straight-line trajectory, with no possible turn. In order to relax this limitation, we are working on extending the reduction presented in Sect. 2.2 to possibly *curved trajectories* of both aircraft.

Another limitation of our approach is that we do not handle intruders equipped with a collision avoidance system, or encounters with multiple intruders. This is made more challenging because those cases are typically handled by online scores *after* interpolation in the score table. But currently, our comparison looks only at the score table and ignores online scores – partly because considering them would add several orders of magnitude to an already large comparison calculation. We would like to extend our model to handle equipped and multiple intruders, and extend our comparison to handle some amount of online scores.

Fewer Counterexamples. The safe regions we identified may be overly conservative – in particular for the COC advisory – which leads to false alarms in the comparison with the actual advisories issued by the ACAS X table. Thus, we may flag an advisory as unsafe, although it is not, simply because of our worst case assumptions. One approach is to improve the precision of our safe regions, to make them less conservative – while keeping them provably safe. Another complementary approach is to sort and categorize the automatically generated counterexamples. This will make it easier for the development team to navigate through the millions of counterexamples we are generating.

Continuous Region Comparisons. So far, the table comparison has been performed at the discrete grid points of the ACAS X table. A natural question one may ask is what happens at the points off this grid? Are the advisories safe for those points? Recall that the ACAS X decisions for off-grid points result from a multi-linear interpolation of the values on the neighboring grid points (vertices). An important fact about the safe regions we identified is that they characterize which actions are safe at any point in the state space regardless of the ACAS X grid. We can thus reuse the same model, proof and safe regions to assess the safety of *any* ACAS X decision.

Although it is conceptually straight-forward to check if one particular point and advisory respect the safe region, checking the safety of an entire region – typically a hyperrectangle – requires first understanding the distribution of optimal ACAS X advisories within the hyperrectangle before checking a geometrical inclusion. Given the large number of cells in the score table, we crucially need a good understanding of the multi-linear interpolation as well as a scalable method to tackle the inclusion problem. For example, whenever all the vertices of the considered hyperrectangle have the same advisory, it is sufficient to only check the safety of the vertices. What is more challenging, however, is when a hyperrectangle exhibits different advisories on its vertices. Figure 6 shows a 2-dimensional projection where the vertices have different optimal advisories. Observe how the multi-linear interpolation computes a different optimal advisory in a non-obvious zone in the middle of the cell.

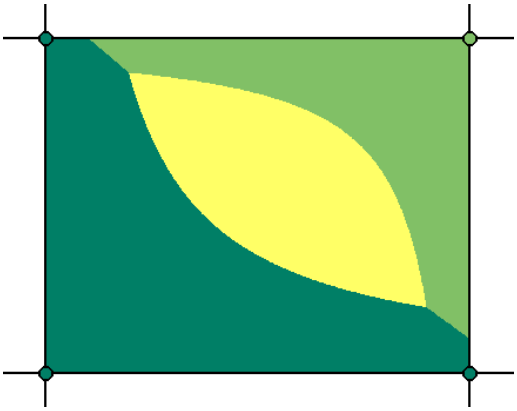


Figure 6: A 2-dimensional example of how interpolation can induce an advisory a to be optimal in the middle of a cell even if a is not optimal on any vertex. The regions in the diagram correspond to areas where different actions are optimal. The color of the circles represents the optimal action at the vertices.

As a first attempt, our strategy to tackle this problem will be as follows. If a cell lies, even partially, in an area where a particular advisory a is unsafe while there exist some points in the cell where a is an optimal advisory, then we may flag these cell as potentially unsafe. This is conservative – the points where a is unsafe may be disjoint from the points where a is optimal. These potentially unsafe cells can be investigated more closely using sampling methods focused on the cell, or further decomposed into smaller cells and recursively analyzed. If, on the other hand, we can prove that no unsafe action can ever be an optimal action within the cell, then we have conclusively proven that the cell is safe under our model. We have a method of answering this cell-checking question by converting it into a sequence of simple convex optimization problems that soundly relax the original question.

Optimality vs. Safety. The ACAS X table is optimized to minimize the probability of a future NMAC while trying to maintain a good operational suitability. In contrast, the regions we identified focus solely on (local) safety aspects for straight-line encounters with no optimality objectives.

How can one get the best of both worlds, i.e., an optimal ACAS X table that is provably safe – at least for the most common encounters? One way of achieving such objective is to integrate the symbolic safe regions into the decisions from the ACAS X table, such that optimal advisories can only be issued if they are deemed safe

with respect to the safe regions. One may also prefer provably safe advisories over ACAS X advisories whenever the safety conditions are violated. We are currently discussing with the development team the following three possible integration schemes:

1. The safe regions could be incorporated into the MDP optimization itself, e.g. by changing the objective function to have a prohibitive score for advisories outside the provable safe regions.

2. As in ModelPlex [14], the final optimal advisory from the ACAS X table could be checked for compatibility with the safe regions and replaced by another safe action if needed.

3. Recall that ACAS X uses *online scores* which adapt the score from the ACAS X table, for example for ground proximity or coordination. The safe regions can be added as a new online score to ACAS X such that it avoids issuing advisories that are not known to be safe.

The above integration propositions feed the knowledge gained from the safety analysis back into the design of ACAS X to help make it an even safer system while preserving its main objective. Approach 1 gives the most direct feedback into the original system optimization but leaves the prohibitive score to use unspecified. Approach 2 is conceptually very clean and comes with strong theoretical guarantees [14] about the final system whenever the assumptions of the hybrid model are met. It requires, however, an additional decision component into the system architecture. Finally, approach 3 leads to a particularly smooth integration, because the ACAS X infrastructure already uses online scores for various purposes on a routine basis. Further evaluation is needed to determine which score to use and how it would interact with other online scores.

Acknowledgments

This research was conducted under the sponsorship of the Federal Aviation Administration Traffic Alert & Collision Avoidance System (TCAS) Program Office (PO) AJM-233 under contract number DTFAWA-11-C-00074. Additionally, support for the basic verification technology used as a foundation for this research was provided by the National Science Foundation under NSF CAREER Award CNS-1054246.

The authors would like to warmly thank Neal Suchy for his lead of the ACAS X project and support of this work, as well as Stefan Mitsch and Jan-David Quesel for their support of the KeYmaera tool. The authors would also like to thank Jeff Brush, Jessica Holland, Robert Klaus, Barbara Kobzik-Juul, Mykel Kochenderfer, Ted Londner, Sarah Loos, Ed Morehouse, Wes Olson, Michael Owen, Joshua Silbermann, and the ACAS X development team for interesting remarks.

5. REFERENCES

[1] B. J. Chludzinski. Evaluation of TCAS II version 7.1 using the FAA fast-time encounter generator model. Technical Report ATC-346, MIT Lincoln Laboratory, April 2009.

[2] G. Dowek, C. Muñoz, and V. Carreño. Provably safe coordinated strategy for distributed conflict resolution. In *AIAA Guidance Navigation, and Control Conference and Exhibit*, 2005.

[3] Federal Aviation Administration. Introduction to TCAS II. Version 7.1, February 2011.

[4] A. Galdino, C. Muñoz, and M. Ayala. Formal verification of an optimal air traffic conflict resolution and recovery algorithm. In *WoLLIC*, volume 4576 of *LNCS*. Springer, 2007.

[5] K. Ghorbal, J.-B. Jeannin, E. Zawadzki, A. Platzer, G. J. Gordon, and P. Capell. Hybrid theorem proving of aerospace systems: Applications and challenges. *Journal of Aerospace Information Systems*, 2014.

[6] J. E. Holland, M. J. Kochenderfer, and W. A. Olson. Optimizing the next generation collision avoidance system for safe, suitable, and acceptable operational performance. *Air Traffic Control Quarterly*, 2014.

[7] J.-B. Jeannin, K. Ghorbal, Y. Kouskoulas, R. Gardner, A. Schmidt, E. Zawadzki, and A. Platzer. A formally verified hybrid system for the next-generation airborne collision avoidance system. Technical Report CMU-CS-14-138, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 2014. KeYmaera files available at `http://www.ls.cs.cmu.edu/pub/acasx.zip`.

[8] J.-B. Jeannin, K. Ghorbal, Y. Kouskoulas, R. Gardner, A. Schmidt, E. Zawadzki, and A. Platzer. A formally verified hybrid system for the next-generation airborne collision avoidance system. In C. Baier and C. Tinelli, editors, *TACAS*, volume 9035 of *LNCS*, pages 21–36. Springer, 2015.

[9] M. J. Kochenderfer and J. P. Chryssanthacopoulos. Robust airborne collision avoidance through dynamic programming. Technical Report ATC-371, MIT Lincoln Laboratory, January 2010.

[10] M. J. Kochenderfer, L. P. Espindle, J. K. Kuchar, and J. D. Griffith. Correlated encounter model for cooperative aircraft in the national airspace system version 1.0. Technical Report ATC-344, MIT Lincoln Laboratory, October 2008.

[11] M. J. Kochenderfer, J. E. Holland, and J. P. Chryssanthacopoulos. Next generation airborne collision avoidance system. *Lincoln Laboratory Journal*, 19(1):17–33, 2012.

[12] S. M. Loos, D. W. Renshaw, and A. Platzer. Formal verification of distributed aircraft controllers. In *HSCC*, pages 125–130. ACM, 2013.

[13] J. Lygeros and N. Lynch. On the formal verification of the TCAS conflict resolution algorithms. In *IEEE Decision and Control*, volume 2, pages 1829–1834. IEEE, 1997.

[14] S. Mitsch and A. Platzer. ModelPlex: Verified runtime validation of verified cyber-physical system models. In B. Bonakdarpour and S. A. Smolka, editors, *RV*, volume 8734 of *LNCS*, pages 199–214. Springer, 2014.

[15] A. Platzer. Differential dynamic logic for hybrid systems. *J. Autom. Reas.*, 41(2):143–189, 2008.

[16] A. Platzer and E. M. Clarke. Formal verification of curved flight collision avoidance maneuvers: A case study. In *FM*, volume 5850 of *LNCS*, pages 547–562. Springer, 2009.

[17] A. Platzer and J.-D. Quesel. KeYmaera: A hybrid theorem prover for hybrid systems. In *IJCAR*, volume 5195 of *LNCS*, pages 171–178. Springer, 2008.

[18] C. Tomlin, G. J. Pappas, and S. Sastry. Conflict resolution for air traffic management: A study in multiagent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):509–521, 1998.

[19] C. von Essen and D. Giannakopoulou. Analyzing the next generation airborne collision avoidance system. In *TACAS*, volume 8413 of *LNCS*, pages 620–635. Springer, 2014.