Unmanned Aircraft Systems in the National Airspace System: A Formal Methods Perspective

César A. Muñoz, NASA
Aaron Dutle, NASA
Anthony Narkawicz, NASA
Jason Upchurch, NASA

As the technological and operational capabilities of unmanned aircraft systems (UAS) have grown, so too have international efforts to integrate UAS into civil airspace. However, one of the major concerns that must be addressed in realizing this integration is that of safety. For example, UAS lack an on-board pilot to comply with the legal requirement that pilots see and avoid other aircraft. This requirement has motivated the development of a detect and avoid (DAA) capability for UAS that provides situational awareness and maneuver guidance to UAS operators to aid them in avoiding and remaining well clear of other aircraft in the airspace. The NASA Langley Research Center Formal Methods group has played a fundamental role in the development of this capability. This article gives a selected survey of the formal methods work conducted in support of the development of a DAA concept for UAS. This work includes specification of low-level and high-level functional requirements, formal verification of algorithms, and rigorous validation of software implementations.

1. INTRODUCTION

In their 2013 economic report, the Association for Unmanned Vehicle Systems International (AUVSI) [Jenkins and Vasigh 2013] projected the cumulative impact of the Unmanned Aircraft Systems (UAS) industry on the US economy between 2015 and 2025 to be more than US $80 billion and the generation of more than one hundred thousand jobs. As the availability of and applications for UAS grow in the US and worldwide, there have been concerted efforts by stakeholders throughout the international community aimed at addressing the problem of safely integrating UAS into standard airspace operations. NASA's Unmanned Aircraft Systems Integration in the National Airspace System (UAS in the NAS) project aims to developing key capabilities to enable routine and safe access for public and civil use of UAS in non-segregated airspace operations.

One of the major challenges to the safe integration of UAS into the NAS is the lack of an on-board pilot to comply with particular US and international legal requirements. In manned aircraft operations on-board pilots have, in part, the responsibility for not "operating an aircraft so close to another aircraft as to create a collision hazard" [International Civil Aviation Organization (ICAO) 2005a; US Code of Federal Regulations 1967a], "to see and avoid other aircraft" [International Civil Aviation Organization (ICAO) 2005b; US Code of Federal Regulations 1967b], and when complying with the particular rules addressing right-of-way, on-board pilots "may not pass over, under, or ahead [of the right-of-way aircraft] unless well clear" [International Civil Aviation Organization (ICAO) 2005b; US Code of Federal Regulations 1967b]. To address the safety challenge and establish parallel requirements for UAS, the final report of the Federal Aviation Administration (FAA) Sense and Avoid (SAA) Workshop [FAA Sponsored Sense and Avoid Workshop 2009] defined the concept of *sense and avoid* as "the capability of a UAS to remain well clear from and avoid collisions with other airborne traffic." This definition has been proposed as a means of compliance with the preceding legal requirements.

In the case of manned aircraft operations, the ability to remain well clear and see and avoid other aircraft depends upon the perception and judgement of the human pilot. In absence of an on-board pilot, there is a need for a formal understanding of the notion of *well-clear*, which resolves this ambiguity and is appropriate for integrated UAS operations [Crück and Lygeros 2007; Coulter 2009; Tomasello and Haddon 2011; Weibel et al. 2011; Adaska 2012; Theunissen et al. 2014; Consiglio et al. 2012]. In recent years, efforts to provide such a definition have been underway.

In 2011, the Office of the Under Secretary of Defense for Acquisition, Technology, and Logistics established the UAS Sense and Avoid Science and Research Panel (SARP) and charged it with making a recommendation for a quantitative definition of well clear. In 2013, the RTCA organization established Special Committee 228 (SC-228) to provide technical guidance to the FAA for defining minimum operational performance standards for a UAS sense and avoid concept, which is now called *detect and avoid*, based on the quantitative definition of well-clear recommended by the UAS SARP. The NASA Langley Research Center (LaRC) Formal Methods group closely collaborated with the UAS SARP to both identify a mathematical definition of well clear that was appropriate for UAS operations and verify that it satisfied operational requirements. The LaRC Formal Methods group is currently participating in the RTCA SC-228, and has responsibility for the specification, development, and verification of a reference implementation of the algorithms that support the overall UAS DAA concept. This article presents an overview of the research undertaken by the LaRC Formal Methods group while collaborating with these groups, and addresses some of the key technical challenges in Formal Methods research in the context of the development and safety analysis of advanced air traffic management concepts.

The mathematical formulas and theorems presented in the subsequent sections have been formally specified and verified in the Prototype Verification System (PVS) [Owre et al. 1992].[1] However, to make the article accessible to a broader audience, the formulas and theorems herein are expressed in mathematical notation instead of concrete PVS syntax. Furthermore, the states of the aircraft pair of interest, denoted *ownship* and *intruder*, are represented by position and velocity vectors in a local East, North, Up (ENU) Cartesian coordinate system. This coordinate system is based on the orthogonal projection of the ownship and intruder geodesic coordinates onto a plane tangent to the projected ownship position on the surface of the earth. For notational convenience, horizontal and vertical components of a three-dimensional vector are represented by a two-dimensional vector and a scalar, respectively, and these components are given in a relative coordinate system, where the intruder is at the origin, and the ownship moves relative to the intruder. Letters in bold-face, e.g., $\mathbf{v}$, $\mathbf{s}$, denote two-dimensional vectors. Finally, vector operations such as addition, subtraction, scalar multiplication, dot product, i.e., $\mathbf{s} \cdot \mathbf{v} \equiv s_x v_x + s_y v_y$, the square of a vector, i.e., $\mathbf{s}^2 \equiv \mathbf{s} \cdot \mathbf{s}$, and the norm of a vector, i.e., $\|\mathbf{s}\| \equiv \sqrt{\mathbf{s}^2}$, are defined in a two-dimensional Euclidean geometry.

2. WELL-CLEAR VOLUME

Consiglio et al. proposed a UAS detect and avoid concept where the well-clear notion is defined by a protected volume around the UAS [Consiglio et al. 2012]. If no traffic aircraft is located inside this volume, the UAS is considered to be well clear. A key idea in this proposal is that the protected volume is assumed to be large enough to avoid resolution advisories from a collision avoidance system, but small enough to avoid disruption to traffic flow. Subsequently, several candidate definitions of the well-clear volume were proposed to the UAS SARP, and they were analyzed with respect to these conceptual requirements as well as other operational requirements [Cook et al. 2015].

The definition of the well-clear volume ultimately recommended by the UAS SARP, and adopted by RTCA SC-228, is a boolean formula based on the second generation

[1] For further information on this formal development, the reader is referred to the directories `TCASII`, `WellClear`, and `DAIDALUS` available in the NASA PVS Library (https://github.com/nasa/pvslib).

of the Traffic Alerting and Collision Avoidance System (TCAS II) Resolution Advisory (RA) detection logic. In particular, the well-clear volume is defined by a predicate on the relative position and velocity vectors of the ownship and intruder at the current time. This predicate determines that two aircraft are *well clear* of each other when computed distance and time functions fall outside a set of predefined threshold values. The particular distance and time functions used in the definition of the well-clear volume are based on those used in the TCAS II RA detection logic [Muñoz et al. 2013] and are discussed in the subsequent presentation.

A well-clear violation is defined as a situation when there is both a horizontal and a vertical violation. That is, the predicate defining the well-clear volume is a conjunction of two predicates, one representing the horizontal dimension and the other representing the vertical dimension, as given in Formula (1).

$$\mathit{WCV}(\mathbf{s}, s_z, \mathbf{v}, v_z) \equiv \mathit{HWCV}(\mathbf{s}, \mathbf{v}) \wedge \mathit{VWCV}(s_z, v_z), \tag{1}$$

Here $\mathbf{s}, \mathbf{v} \in \mathbb{R}^2$ are the respective relative horizontal position and velocity vectors of the aircraft pair, and $s_z, v_z \in \mathbb{R}$ are the respective relative vertical position and velocity of the aircraft pair. The horizontal and vertical violation predicates are defined in Formula (2) and Formula (3), respectively.

$$\mathit{HWCV}(\mathbf{s}, \mathbf{v}) \equiv \|\mathbf{s}\| \leq \mathtt{DMOD} \vee (\mathit{HMDF}(\mathbf{s}, \mathbf{v}) \wedge 0 \leq \tau_{\text{mod}}(\mathbf{s}, \mathbf{v}) \leq \mathtt{TAUMOD}), \tag{2}$$

$$\mathit{VWCV}(s_z, v_z) \equiv |s_z| \leq \mathtt{ZTHR} \vee 0 \leq t_{\text{coa}}(s_z, v_z) \leq \mathtt{TCOA}, \tag{3}$$

where TAUMOD and DMOD are horizontal time and distance thresholds, respectively, and TCOA and ZTHR are vertical time and distance thresholds, respectively. The predicate *HMDF* refers to the *horizontal miss-distance filter* and is defined as in Formula (4).

$$\mathit{HMDF}(\mathbf{s}, \mathbf{v}) \equiv d_{\text{cpa}}(\mathbf{s}, \mathbf{v}) \leq \mathtt{HMD}, \tag{4}$$

where HMD is the horizontal miss-distance threshold and is typically set to the same value as DMOD. The distance function d_{cpa} computes the projected horizontal distance between the aircraft at their closest point of approach in the horizontal dimension, assuming constant relative horizontal velocity $\mathbf{v}$, and is formally defined in Formula (5).

$$d_{\text{cpa}}(\mathbf{s}, \mathbf{v}) \equiv \|\mathbf{s} + t_{\text{cpa}}(\mathbf{s}, \mathbf{v})\mathbf{v}\|. \tag{5}$$

The time function t_{cpa} in Formula (5) is the time to closest point of approach, and is defined as

$$t_{\text{cpa}}(\mathbf{s}, \mathbf{v}) \equiv \begin{cases} -\frac{\mathbf{s} \cdot \mathbf{v}}{\mathbf{v}^2} & \text{if } \|\mathbf{v}\| \neq 0, \\ 0 & \text{otherwise}, \end{cases} \tag{6}$$

where the inequality $\mathbf{s} \cdot \mathbf{v} < 0$ holds when the aircraft are horizontally converging, $\mathbf{s} \cdot \mathbf{v} > 0$ holds when the aircraft are horizontally diverging, and $\mathbf{s} \cdot \mathbf{v}$ equals 0 when the aircraft are at their horizontal closest point of approach.

The time function τ_{mod} (*modified tau*) was introduced in the TCAS II RA logic [Hammer 1996]. In the vector notation used in this article, modified tau is defined in Formula (7).

$$\tau_{\text{mod}}(\mathbf{s}, \mathbf{v}) \equiv \begin{cases} \frac{\mathtt{DMOD}^2 - \mathbf{s}^2}{\mathbf{s} \cdot \mathbf{v}} & \text{if } \mathbf{s} \cdot \mathbf{v} < 0, \\ -1 & \text{otherwise}. \end{cases} \tag{7}$$

The time function t_{coa} computes the time to co-altitude, assuming constant relative vertical speed v_z, and is defined in Formula (8).

$$t_{\text{coa}}(s_z, v_z) \equiv \begin{cases} -\frac{s_z}{v_z} & \text{if } s_z v_z < 0, \\ -1 & \text{otherwise}. \end{cases} \tag{8}$$

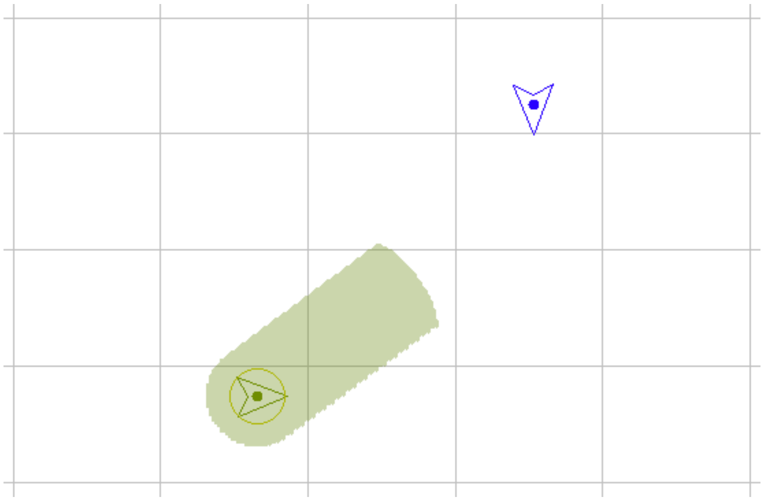

Fig. 1. Top View of Well-Clear Volume

In a way similar to the horizontal case, the product $s_z v_z$ characterizes whether the aircraft are vertically diverging, i.e., $s_z v_z > 0$, or vertically converging, i.e., $s_z v_z < 0$. For completeness, time to co-altitude is defined as -1 when the aircraft are not vertically converging.

The values of the distance and time thresholds recommended by the UAS SARP [Cook et al. 2015] and adopted by the RTCA SC-228 are $\texttt{DMOD} = \texttt{HMD} = 4000$ ft, $\texttt{ZTHR} = 450$ ft, $\texttt{TAUMOD} = 35$ s, and $\texttt{TCOA} = 0$ s. The shaded area in Figure 1 illustrates the resultant top view of the ownship's well-clear volume with respect to an intruder aircraft for an example encounter. Any traffic aircraft located inside this area with the same velocity vector as the intruder aircraft will be in well-clear violation with the UAS. The diameter of the circular base of the well-clear volume is DMOD. The length of the elongated shape depends on TAUMOD. The shape is elongated in the direction of the relative velocity vector.

For arbitrary values of DMOD, ZTHR, TAUMOD, and TCOA, with $\texttt{HMD} = \texttt{DMOD}$, Formula (1) satisfies several operational requirements [Muñoz et al. 2014]. For instance, in a pairwise encounter, the ownship and intruder aircraft make the same determination of their well-clear status. That, both aircraft are simultaneously aware of an in agreement on a well-clear violation. This property is called *symmetry*, and it is formally stated in Theorem 2.1.

THEOREM 2.1 (SYMMETRY). *For all relative states* $\mathbf{s}, s_z, \mathbf{v}, v_z$*,* $\mathit{WCV}(\mathbf{s}, s_z, \mathbf{v}, v_z) = \mathit{WCV}(-\mathbf{s}, -s_z, -\mathbf{v}, -v_z)$.

Another important property states that for straight line trajectories there is at most one time interval where the aircraft are not well clear. This property is called *local convexity*, and it enables the definition of an alerting algorithm that, in a non-maneuvering encounter, continuously alerts a predicted well-clear violation until the violation disappears. Once the violation disappears, it does not reappear unless the aircraft maneuver. This property is formally stated in Theorem 2.2.

THEOREM 2.2 (LOCAL CONVEXITY). *For all relative states* $\mathbf{s}, s_z, \mathbf{v}, v_z$*, if there are times* $0 \leq t_1 \leq t_2$ *such that* $\mathit{WCV}(\mathbf{s}+t_1\mathbf{v}, s_z+t_1 v_z, \mathbf{v}, v_z)$ *and* $\mathit{WCV}(\mathbf{s}+t_2\mathbf{v}, s_z+t_2 v_z, \mathbf{v}, v_z)$*, then for all times* $t1 \leq t \leq t2$*,* $\mathit{WCV}(\mathbf{s}+t\mathbf{v}, s_z+t v_z, \mathbf{v}, v_z)$.

The well-clear volume defined by Formula (1) assumes perfect aircraft state information. To accomodate for uncertainty in the position and velocity information, the RTCA SC-228 requirements for the well-clear alerting logic allows for the use of a larger set of threshold values within some specified ranges, giving an *extended well-clear volume*. This extended well-clear volume is characterized by a predicate *WCV** defined by Formula (1), using parameters $\texttt{DMOD}^* \geq \texttt{DMOD}$, $\texttt{HMD}^* \geq \texttt{HMD}$, $\texttt{ZTHR}^* \geq \texttt{ZTHR}$, $\texttt{TAUMOD}^* \geq \texttt{TAUMOD}$, and $\texttt{TCOA}^* \geq \texttt{TCOA}$. Theorem 2.3 formally states that the well-clear volume instantiated with the RTCA SC-228 standard threshold values is safely included in any of its extensions.

THEOREM 2.3 (EXTENSION). *For all relative states* $\mathbf{s}, s_z, \mathbf{v}, v_z$, *WCV*$(\mathbf{s}, s_z, \mathbf{v}, v_z) \implies$ *WCV*$^*(\mathbf{s}, s_z, \mathbf{v}, v_z)$.

The extension property enables the definition of formally verified alerting and maneuvering algorithms that *completely* protect against the violation of the standard well-clear volume by using a more conservative definition.

3. WELL-CLEAR AND TCAS II INTEROPERABILITY

TCAS is a family of airborne devices that are designed to reduce the risk of mid-air collisions between aircraft equipped with operating transponders. TCAS II [RTCA SC-147 2009], the current generation of TCAS devices, is mandated in the US for aircraft with greater than 30 seats or a maximum takeoff weight greater than 33,000 pounds. Although not required, TCAS II is also installed on many turbine-powered general aviation aircraft. TCAS II provides *resolution advisories* (RAs), which are visual and vocalized alerts that direct pilots to maintain or increase vertical separation with intruders that are considered collision threats.

Formula (1), which defines the UAS well-clear volume, closely follows the formula that defines the logic of the TCAS II RA detection algorithm. The primary difference between the well-clear volume, as defined by RTCA SC-228, and the TCAS II RA volume is in the particular choice of time and distance threshold values. The TCAS II RA detection logic uses a lookup table indexed by the altitude of the ownship to determine which set of threshold values to use. For some entries, e.g., for aircraft flying above 10,000 ft, the TCAS II RA threshold values are higher than the well-clear threshold values. Due to these particular instances of larger threshold values for TCAS II, the well-clear volume is not a proper extension of the TCAS II RA volume. Hence, Theorem 2.3 does not apply to these cases. Thus, in practice, it is possible that TCAS II issues an RA before the ownship declares a well-clear violation. The vertical threshold values in the well-clear definition are also problematic. Indeed, $\texttt{ZTHR}$ and $\texttt{TCOA}$ for the well-clear volume are set to 450 ft and 0 s, respectively, and these values are always smaller than the corresponding threshold values in the TCAS II RA table. This choice of values affects the interoperability of the UAS DAA concept with TCAS II in encounters with high vertical closure rate [Upchurch et al. 2015].

There are operational justifications for this mismatch between the well-clear definition and the TCAS II RA threshold values. For instance, the detect and avoid concept considered by RTCA SC-228 only applies to certain types of UAS and in classes of airspace that are usually below 10,000 ft, that is, Class D, Class E, and perhaps Class G airspace. Additionally, the negative effects of this mismatch, i.e., TCAS II RAs preceding well-clear violations, may be mitigated by the use of extended well-clear volumes in the alerting and maneuvering algorithms, which are also part of the RTCA SC-228 UAS detect and avoid concept. Theorem 3.1 states that there are extended well clear volumes that properly include the TCAS II RA volume.

THEOREM 3.1 (INCLUSION). *Let* $\mathit{HMD}^* = \mathit{DMOD}^*$, $\mathit{TAUMOD}^* = \mathit{TCOA}^*$, *and* ZTHR^* *be threshold values larger than the corresponding TCAS II RA threshold values. For all relative states* $\mathbf{s}, s_z, \mathbf{v}, v_z$, $\mathit{TCASII\_RA}(\mathbf{s}, s_z, \mathbf{v}, v_z) \implies \mathit{WCV}^*(\mathbf{s}, s_z, \mathbf{v}, v_z)$.

Despite Theorem 3.1, it has been observed in flight tests that it is still possible for TCAS II to issue an RA before a violation of an extended well-clear volume. The reason for this apparent inconsistency is that Theorem 3.1 assumes that the same state information, in vector form, is available to both systems, yet this is not the case. In the case of the well-clear concept, state information in vector form is readily available through modern global positioning systems. On the other hand, TCAS II assumes that aircraft are equipped with active transponders, and the lack of reliable vector information is compensated for by a sophisticated tracking system. This tracking system is a key component of the TCAS II RA system, and depending on the quality of the range rate estimate computed by the tracker and other conditions, the TCAS II RA system may disable the use of the horizontal miss distance filter in Formula (2), i.e., $\mathit{HMDF}(\mathbf{s}, \mathbf{v})$ is considered to be true, leading to the case of TCAS II RAs preceding violation of the extended well-clear volume.

4. DAIDALUS

The RTCA SC-228 Minimum Operational Performance Standards (MOPS) for Unmanned Aircraft Systems includes a reference implementation of DAA algorithms that assist remote pilots by providing situational awareness of proximity to other aircraft in the airspace. This reference implementation, called DAIDALUS (Detect & Avoid Alerting Logic for Unmanned Systems [Muñoz et al. 2015], is being developed by the LaRC Formal Methods group in support of NASA's Safe Autonomous System Operations (SASO) project. The DAIDALUS source code is available in both C++ and Java under NASA's Open Source Agreement[2].

The algorithms contained in DAIDALUS compute: (1) predictions of well-clear violations between the ownship and a given intruder aircraft, (2) maneuver guidance for the ownship to maintain or regain well-clear status with respect to all traffic aircraft, and (3) an alert level representing the severity of a potential well-clear violation between the ownship and a given intruder aircraft.

A fundamental element of DAIDALUS is the algorithm that computes the time interval of violation for a non-maneuvering encounter. This algorithm, called `wc_interval`, has as inputs a relative aircraft state and a non-empty lookahead time interval $[B, T]$, with $0 \leq B < T$. It returns the time interval $[t_{\text{in}}, t_{\text{out}}]$, which corresponding to the interval of a predicted well-clear violation, assuming constant-velocity trajectories. The returned interval is empty if no such violation is predicted. Theorem 4.1 states that `wc_interval` is correct and complete.

THEOREM 4.1 (CORRECT AND COMPLETE). *For all relative states* $\mathbf{s}, s_z, \mathbf{v}, v_z$ *and times* B, T, *with* $0 \leq B < T$, *let* $[t_{in}, t_{out}]$ *be the time interval returned by* $\mathit{wc\_interval}(\mathbf{s}, s_z, \mathbf{v}, v_z, B, T)$ *and* t *be a time in* $[B, T]$, *then* $\mathit{WCV}(\mathbf{s} + t\mathbf{v}, s_z + tv_z, \mathbf{v}, v_z)$ *if and only if* $t \in [t_{in}, t_{out}]$.

The local convexity property given by Theorem 2.2 is a necessary condition for the existence of an algorithm, such as `wc_interval`, that satisfies Theorem 4.1. Without local convexity, either one of the implications in the theorem must be removed.

The algorithms that comprise DAIDALUS are specified in PVS, and a large number of their functional requirements, such as Theorem 4.1, are proven to hold in these formal models of the algorithms. The DAIDALUS algorithms are implemented in Java

[2] http://github.com/nasa/wellclear.

and C++, and the formal verification of these implementations, while possible in some cases, is exceedingly difficult. The difficulty arises due to the use of floating point arithmetic in the implementations, while the formal verification of these algorithms is done over the real numbers. This difficulty is also partially due to the object-oriented nature of these languages versus the functional PVS notation. The difficulty is further compounded when, as in the case of DAIDALUS, the properties to be verified are themselves complex.

The validation of the DAIDALUS software implementation is performed using a more pragmatic approach named *model animation* [Dutle et al. 2015]. While model animation does not offer the same high level of safety assurance as a complete formal verification, it addresses the numerical issues in practical way, while still offering strong assurance that the implemented software performs to its specification. Three elements are required in order to implement the technique: (1) there must be an executable formal model of the software, which has been verified to possess the desired properties of the actual software, (2) the software to be assessed needs to be a close translation from the formal specification into the desired language, and (3) a collection of representative test cases must be selected in some manner decided by the user. Each test case is then evaluated using the formal model and the software implementation, and the results are compared. If the results are sufficiently close for an acceptable number of the test cases, then the software is considered to faithfully implement the formal model. However, there are details which present technical challenges in the model animation process.

The algorithms in DAIDALUS use mathematical functions such as roots, trigonometric, and inverse trigonometric functions. Due to the presence of irrational and transcendental numbers, the outputs computed by these algorithms are not exact. Nonetheless, other than the presence some of these non-computable functions, the formal specifications of the DAIDALUS algorithms are executable using the PVS ground evaluator. In order to fully execute these algorithms, PVS includes an animation tool called PVSio [Muñoz 2003], which extends the ground evaluator, notably by providing support for *semantic attachments*; a semantic attachment is a way to replace a PVS function call with a call to a trusted oracle. In its default mode, PVSio replaces non-computable mathematical functions with internal LISP floating point functions. When more precision is needed, or when floating point evaluation must be avoided, these standard functions are replaced with a collection of attachments that are formally verified in PVS to be correct up to a given, but arbitrary, precision.

Other fundamental model animation design decisions relate to the selection of a proper collection of test cases, how to determine when outputs are "sufficiently close," and how many disagreements between the software and the model are acceptable. The selection of a set of test cases for this method can be done in any number of ways. Test cases might be selected to ensure some code coverage criteria are met, or a large number of random cases may be chosen to allow for a wide variety of inputs to be generated, or other desired criteria may go into the selection process. In the case of assessing the DAA functionality of DAIDALUS, the test set chosen began as a collection of 95 aircraft encounter scenarios, which were developed jointly by the USAF, MIT Lincoln Laboratory, and NASA during the UAS SARP activity. These scenarios represent a suite of stressing cases which were designed to be difficult cases for the well-clear logic. Throughout the DAIDALUS development and assessment process, a number of additional stressing scenarios have been provided by the FAA, and others, and added to the test set.

For some of the functionality provided by DAIDALUS, testing whether the formal models and their reference implementations agree or not is straightforward. For example, the well-clear violation logic computes a boolean value indicating the well-clear status between the ownship and an intruder aircraft. Testing if this logic is correctly implemented in software amounts to checking if the same boolean value is computed in both the formal model and the software implementation. However, for algorithms that compute a numerical value, the verification is less straightforward. In such cases, even very close outputs might be different due to numerical approximations. For these numerical differences, an allowable tolerance for each particular output is determined. If the difference between the formal model and the implementation is within this tolerance, then the two are declared to be in agreement. These tolerances are determined on a case-by-case basis, based on the needed accuracy of the output. For example, if the output is a time value intended to be displayed to an operator in whole number seconds then a tolerance for such a value might be set to 0.5 seconds. Furthermore, if the output is a distance based on ADS-B, then a tolerance of one meter might be sufficient, since the guaranteed accuracy of ADS-B is within approximately five meters. The choices made for these tolerances also directly affect the number of cases in which the software and formal model disagree, which is the final criterion for determining whether or not they agree, overall.

In the case of DAIDALUS, this validation procedure is being performed in conjunction with the development of the software. As such, the procedure not only validates that the software is in agreement with its formal model, but also reveals places where the software must be modified to be in closer agreement with its formal model. At each step in this iterative process, the DAIDALUS software implementation is brought closer to matching its formal model in all of its functionality, thus providing a high level of assurance that the implemented software retains the safety properties proven in the formal model.

5. CONCLUSION

Motivated by the safety-critical nature of a detect and avoid concept and the corresponding need for strong assurances and mathematical guarantees, Formal Methods research has contributed to the formal development of a DAA concept for UAS. The use of Formal Methods in the DAA problem includes a formal definition of the well-clear violation volume, formal proofs of its key properties, formal specification and verification of DAA algorithms, and the rigorous validation of the software implementation of these algorithms against their formal specifications. All formal specifications and proofs supporting this work are written and mechanically verified in the interactive theorem prover, PVS.

The application of Formal Methods to the safety analysis of air traffic management systems faces technical challenges common to complex cyber-physical systems (CPS). Chief among these challenges is the interaction of these systems with the physical environment that yields mathematical models with both continuous and discrete behaviors. Formally proving properties involving continuous mathematics, such non-linear arithmetic, in particular, is a well-known problem in automated deduction. As part of this research effort, several automated decision and semi-decision procedures for dealing with different kinds of non-linear real arithmetic problems have been developed [Narkawicz et al. 2015; Moscato et al. 2015; Denman and Muñoz 2014; Narkawicz and Muñoz 2014; Muñoz and Narkawicz 2013]. Most of these procedures are formally verified and are available as proof-producing automated strategies in the PVS theorem prover.

The formal verification of software implementations of a CPS is a major endeavor, even when the algorithms that are implemented have been formally verified. The main difficulty arises from the fact that modern programming languages use floating point arithmetic, while formal verification is usually performed over the real numbers. Furthermore, there is a large semantic gap between modern programming languages and the functional notation used in formal tools such as PVS. In the research discussed in this paper, model animation is used as a practical approach to the validation of numerical software. Model animation compares computations performed in the software implementations against those symbolically evaluated to an arbitrary precision on the corresponding formal models. While this approach does not provide an absolute guarantee that the software is correct, it increases the confidence that the formal models are faithfully implemented in code.

Finally, air traffic management systems are unique in many aspects. For instance, these systems involve human and automated elements and these elements are often subject to strict operational and legal requirements. These requirements restrict the design space of operational concepts, such as DAA for UAS. More importantly, new concepts and algorithms have to support an incremental evolution of the airspace system on a global scale. Thus, solutions may result which are non-optimal from a theoretical point of view, or which may have complex verification issues due to legacy systems such as TCAS. This article has presented a survey of Formal Methods research and applications to the practical problem of safely integrating UAS into the NAS, a design space thus constrained by requirements for safety, interoperability with legacy systems, regulatory compliance, and public trust.

ACKNOWLEDGMENTS

The work presented in this paper was conducted in support of the Unmanned Aircraft Systems Integration in the National Airspace System project and the Safe Autonomous System Operations project at NASA.

REFERENCES

Jason Adaska. 2012. Computing risk for Unmanned Aircraft self separation with maneuvering intruders. In *Proceedings of the 31st IEEE/AIAA Digital Avionics Systems Conference (DASC)*. 8A4–1–8A4–10.

María Consiglio, James Chamberlain, César Muñoz, and Keith Hoffler. 2012. Concept of integration for UAS operations in the NAS. In *Proceedings of 28th International Congress of the Aeronautical Sciences, ICAS 2012*. Brisbane, Australia.

Stephen P. Cook, Dallas Brooks, Rodney Cole, Davis Hackenberg, and Vincent Raska. 2015. Defining Well Clear for Unmanned Aircraft Systems. In *Proceedings of the 2015 AIAA Infotech @Aerospace Conference*. Kissimmee, Florida.

Dennis Coulter. 2009. UAS integration into the national airspace system: modeling the sense and avoid challenge. In *Proceedings of the 2009 AIAA Infotech at Aerospace Conference*. Seattle, Washington.

Eva Crück and John Lygeros. 2007. Sense and avoid system for a MALE UAV. In *Proceedings of the 2007 AIAA Conference on Guidance, Navigation and Control*. Hilton Head, SC.

William Denman and César Muñoz. 2014. Automated Real Proving in PVS via MetiTarski. In *Proceedings of the 19th International Symposium on Formal Methods (FM 2014) (Lecture Notes in Computer Science)*, Cliff Jones, Pekka Pihlajasaari, and Jun Sun (Eds.), Vol. 8442. Springer, Singapore, 194–199.

Aaron Dutle, César Muñoz, Anthony Narkawicz, and Ricky Butler. 2015. Software Validation via Model Animation. In *Proceedings of the 9th International Conference on Tests & Proofs (TAP 2015) (Lecture Notes in Computer Science)*, Jasmin Blanchette and Nikolai Kosmatov (Eds.), Vol. 9154. Springer, L'Aquila, Italy, 92–108. DOI:http://dx.doi.org/10.1007/978-3-319-21215-9_6

FAA Sponsored Sense and Avoid Workshop. 2009. Sense and avoid (SAA) for Unmanned Aircraft Systems (UAS). (October 2009).

Jonathan Hammer. 1996. Horizontal miss distance filter system for suppressing false resolution alerts. (October 1996). U.S. Patent 5,566,074.

International Civil Aviation Organization (ICAO). 2005a. Annex 2 to the Convention on International Civil Aviation. (July 2005).

International Civil Aviation Organization (ICAO). 2005b. Annex 2 to the Convention on International Civil Aviation. (July 2005).

Darryl Jenkins and Bijan Vasigh. 2013. The economic impact of Unmanned Aircraft Systems integration in the United States. Economic report of the Association For Unmanned Vehicle Systems International (AUVSI). (March 2013).

Mariano Moscato, César Muñoz, and Andrew Smith. 2015. Affine Arithmetic and Applications to Real-Number Proving. In *Proceedings of the 6th International Conference on Interactive Theorem Proving (ITP 2015) (Lecture Notes in Computer Science)*, Christian Urban and Xingyuan Zhang (Eds.), Vol. 9236. Springer, Nanjing, China.

César Muñoz. 2003. *Rapid prototyping in PVS*. Contractor Report NASA/CR-2003-212418. NASA, Langley Research Center, Hampton VA 23681-2199, USA.

César Muñoz and Anthony Narkawicz. 2013. Formalization of a Representation of Bernstein Polynomials and Applications to Global Optimization. *Journal of Automated Reasoning* 51, 2 (August 2013), 151–196. DOI:http://dx.doi.org/10.1007/s10817-012-9256-3

César Muñoz, Anthony Narkawicz, and James Chamberlain. 2013. A TCAS-II Resolution Advisory Detection Algorithm. In *Proceedings of the AIAA Guidance Navigation, and Control Conference and Exhibit 2013*. Boston, Massachusetts.

César Muñoz, Anthony Narkawicz, James Chamberlain, María Consiglio, and Jason Upchurch. 2014. A Family of Well-Clear Boundary Models for the Integration of UAS in the NAS. In *Proceedings of the 14th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*. Atlanta, Georgia, USA.

César Muñoz, Anthony Narkawicz, George Hagen, Jason Upchurch, Aaron Dutle, and María Consiglio. 2015. DAIDALUS: Detect and Avoid Alerting Logic for Unmanned Systems. In *Proceedings of the 34th Digital Avionics Systems Conference (DASC 2015)*. Prague, Czech Republic.

Anthony Narkawicz and César Muñoz. 2014. A Formally Verified Generic Branching Algorithm for Global Optimization. In *Proceedings of the 5th International Conference on Verified Software: Theories, Tools, and Experiments (VSTTE 2013) (Lecture Notes in Computer Science)*, Ernie Cohen and Andrey Rybalchenko (Eds.), Vol. 8164. Springer, Menlo Park, CA, US, 326–343.

Anthony Narkawicz, César Muñoz, and Aaron Dutle. 2015. Formally-Verified Decision Procedures for Univariate Polynomial Computation Based on Sturm's and Tarski's Theorems. *Journal of Automated Reasoning* 54, 4 (2015), 285–326. DOI:http://dx.doi.org/10.1007/s10817-015-9320-x

Sam Owre, John Rushby, and Natarajan Shankar. 1992. PVS: A Prototype Verification System. In *Proceedings of the 11th International Conference on Automated Deduction (Lecture Notes in Artificial Intelligence)*, Deepak Kapur (Ed.), Vol. 607. Springer-Verlag, 748–752.

RTCA SC-147. 2009. RTCA-DO-185B, Minimum operational performance standards for traffic alert and collision avoidance system II (TCAS II). (July 2009).

Erik Theunissen, Brandon Suarez, and Maarten Uijt de Haag. 2014. The impact of a quantitative specification of a Well Clear Boundary on pilot displays for self separation. In *Proceedings of the 2014 Integrated Communications, Navigation and Surveillance Conference (ICNS)*. IEEE, F2–1 – F2–11.

Filippo Tomasello and David Haddon. 2011. Detect and avoid for Unmanned Aircraft Systems in the total system approach. In *Proceedings of the 2011 Tyrrhenian International Workshop on Digital Communications-Enhanced Surveillance of Aircraft and Vehicles (TIWDC/ESAV)*. IEEE, Capri, Italy, 47–52.

Jason Upchurch, César Muñoz, Anthony Narkawicz, María Consiglio, and James Chamberlain. 2015. Characterizing the Effects of a Vertical Time Threshold for a Class of Well-Clear Definitions. In *Proceedings of the 11th USA/Europe Air Traffic Management R&D Seminar, ATM 2015*. Lisbon, Portugal.

US Code of Federal Regulations. 1967a. Title 14 Aeronautics and Space; Part 91 General operating and fight rules. (1967).

US Code of Federal Regulations. 1967b. Title 14 Aeronautics and Space; Part 91 General operating and fight rules. (1967).

Roland E Weibel, Matthew WM Edwards, and Caroline S Fernandes. 2011. Establishing a risk-based separation standard for unmanned aircraft self separation. In *Proceedings of the Ninth USA/Europe Air Traffic Management Research & Development Seminar*. Berlin, Germany.