Sensor Uncertainty Mitigation and Dynamic Well Clear Volumes in DAIDALUS

Anthony Narkawicz
NASA Langley Research Center
Hampton, VA

César Muñoz
NASA Langley Research Center
Hampton, VA

Aaron Dutle
NASA Langley Research Center
Hampton, VA

***Abstract*—This paper presents recent updates to DAIDALUS (Detect & Avoid Alerting Logic for Unmanned Systems), a *detect and avoid* (DAA) software package for the integration of civil UAS into the airspace. DAIDALUS is the reference implementation of detect and avoid for unmanned aircraft systems chosen by RTCA Special Committee 228 (SC-228), and it is included in its corresponding Minimum Operational Performance Standards (MOPS) document, DO-365. This paper reports on the integration into DAIDALUS of two new capabilities, namely dynamic well clear volumes and sensor uncertainty mitigation.[1]**

***Index Terms*—DAIDALUS, well clear, unmanned, detect and avoid**

I. Introduction

In support of requirements developed by RTCA Special Committee 228 (SC-228) for unmanned aircraft in the national airspace, NASA has developed a *detect and avoid (DAA)* concept for UAS [1]. The NASA DAA concept includes an open-source[2] suite of algorithms, implemented in Java and C++, called DAIDALUS (Detect and Avoid Alerting Logic for Unmanned Systems) [2]. The top-level functionality provided by DAIDALUS is situational awareness to UAS operators in the form of alerts and maneuver guidance intended to aid in maintaining well-clear status and recovering well-clear status if it is lost. DAIDALUS is the reference implementation of detect and avoid for unmanned aircraft systems chosen by SC-228, and it is included in its corresponding Minimum Operational Performance Standards (MOPS) document, DO-365.

At the core of the DAA concept implemented in DAIDALUS, there is a mathematical definition of the well-clear concept for unmanned systems. Two aircraft are considered to be *well clear* if appropriate distance and time variables determined by the relative aircraft states remain outside a set of predefined threshold values. This logic is based on the Resolution Advisory (RA) logic of the Traffic Alert and Collision Avoidance System Version II (TCAS II). DAIDALUS includes algorithms for determining the current well-clear status between two aircraft, predicting a loss of well-clear within a lookahead time, and computing the time interval of well-clear violation. These algorithms can be configured to allow for many different sized well-clear volumes to be defined. It also provides algorithms for computing separation assurance bands, assuming a simple kinematic trajectory model. These bands are ranges of track, ground speed, vertical speed, and altitude maneuvers that are predicted to cause a loss of well-clear within a given lookahead time. When aircraft are not well clear, or when a loss of well-clear is unavoidable, the DAIDALUS bands algorithms compute well-clear recovery bands, which give ranges of horizontal and vertical maneuvers that assist pilots in regaining well-clear status. Finally, DAIDALUS provides an alerting scheme that computes a numerical value indicating the severity of a potential loss of well-clear.

This paper reports on the integration into DAIDALUS of the following *two new capabilities*, which will be available in DAIDALUS v2.0.

- dynamic well clear volumes, and
- sensor uncertainty mitigation.

DAIDALUS version 1.0 uses an alerting and guidance scheme based on a single definition of well-clear for all aircraft. By default, this is the definition chosen by SC-228 in DO-365. Future changes to the well-clear concept may include different well-clear definitions based on whether aircraft are in a terminal area or meet some other criteria. This paper describes the recent addition to DAIDALUS of *dynamic well-clear volumes*, which provides the capability of defining alerting and guidance schemes based on different well-clear definition parameters for different aircraft, and to change these schemes on-the-fly. In particular, DAIDALUS now allows multiple instances of the *alerter* class, which allows the definition of well-clear and the associated alerting and guidance parameters to be defined (and even changed) dynamically. An *ownship* aircraft can choose a specific alerter and corresponding well clear definition for each *intruder* aircraft, or change the alerter for all intruders if the ownship enters a different phase of flight.

This paper also describes new functions in DAIDALUS for *sensor uncertainty mitigation*, which detect and predict well-clear violations in the presence of sensor uncertainties. These algorithms represent a substantial addition to DAIDALUS, in both capability and complexity, although these changes are mostly back-end changes and do not affect the user interface. These sensor uncertainty mitigation algorithms take as inputs the uncertainties in aircraft positions and velocities, typically described by variances and covariances on their east-north-

[1] This draft corrects a typo in the published version in page 5, where the symbols $>$ and $\leq$ where exchanged.

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

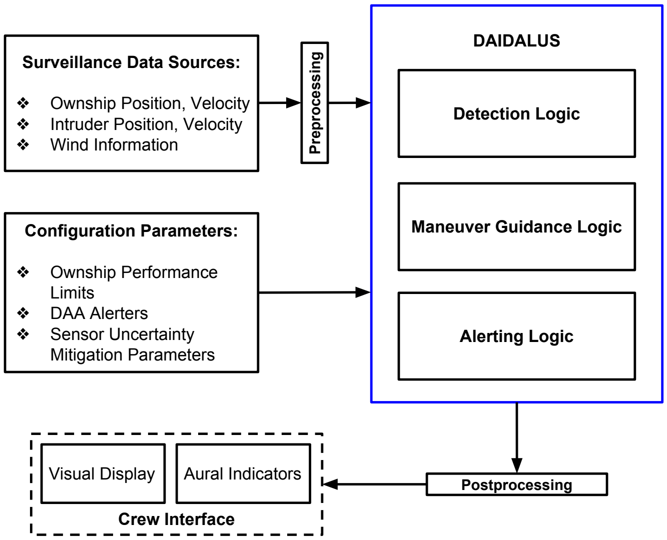
Surveillance Data Sources:
❖ Ownship Position, Velocity
❖ Intruder Position, Velocity
❖ Wind Information
Preprocessing
DAIDALUS
Detection Logic
Maneuver Guidance Logic
Alerting Logic
Configuration Parameters:
❖ Ownship Performance Limits
❖ DAA Alerters
❖ Sensor Uncertainty Mitigation Parameters
Visual Display
Aural Indicators
Crew Interface
Postprocessing

Fig. 1. High-Level Architecture of DAIDALUS

up components. The paper presents these algorithms, as well as the formal verification of the algorithms in an *interactive theorem prover*, which provides formally defined models and correctness properties that serve as documentation for the algorithms' behavior. The interactive theorem prover used in the DAIDALUS development is the Prototype Verification System (PVS) [3].

The algorithms for sensor uncertainty mitigation are unique in that they are based on *formally proven* mathematical formulas that guarantee bounds on the probability of violation (well-clear violation, alerting, etc.) given certain assumptions on the trajectories of the aircraft. This approach stands in contrast to standard methods for incorporating uncertainty information into airspace system that are based on setting parameters to ensure appropriate simulation performance. In particular, the approach taken with these new algorithms in DAIDALUS are based on an explicit mathematical derivation of the impact of uncertainty on the functions defining violations in DAIDALUS and the SC-228 MOPS standards in DO-365.

II. Overview of DAIDALUS

This section reviews the basic functionality of DAIDALUS. Further information can be found in [2]. DAIDALUS is a collection of algorithms that is intended to satisfy the operational and functional requirements detailed in the Minimum Operational Performance Standards (MOPS) document, DO-365, which is produced by RTCA SC-228. The high-level functional relationship between the DAIDALUS implementation and the surveillance data sources, separation standards, and crew interface is depicted in Figure 1, as in [2].

In particular, DAIDALUS provides algorithms that:

1) determine the current, pairwise well-clear status of the ownship and all aircraft inside its surveillance range,
2) compute maneuver guidance in the form of ranges of maneuvers that a pilot-in-command (PIC) may take that will cause the aircraft to maintain or increase separation from the well-clear violation volume, or allow for recovery from loss of well-clear status in a timely manner within the performance limits of the ownship aircraft, and
3) determine the corresponding alert type, based on a given alerting schema, corresponding to the level of threat to the well-clear volume.

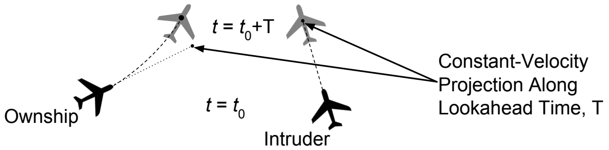

Fig. 2. Constant Velocity Aircraft Projection

The functionalities provided in 1), 2), and 3) are respectively referred to as *detection logic*, *maneuver guidance logic*, and *alerting logic*, as illustrated in Figure 1.

DAIDALUS provides algorithms to compute well-clear information for two aircraft, the *ownship* and the *intruder*. The detection logic computes a predicted time interval of well-clear violation, using linear projections of their states, which are illustrated in Figure 2.

The maneuver guidance provided by DAIDALUS is presented in the form of *separation assurance bands*, i.e., ranges of ownship maneuvers that either avoid or lead to a well-clear violation, and *recovery bands*, i.e., ranges of ownship maneuvers that recover from a present or unavoidable well-clear violation. Bands in DAIDALUS are provided corresponding to four types of aircraft maneuvers: (1) track ranges (or heading, if wind information is provided), (2) ground speed ranges (or air speed, if wind information is provided), (3) vertical speed ranges, (4) and altitude ranges (with a climb rate specified by the user). Figure 3 illustrates state projections for the ownship and intruder aircraft used in the computation of track separation assurance bands. Bands in DAIDALUS are further classified as either *conflict* or *peripheral*. A band is called a conflict band if the band includes the ownship's current velocity vector. That is, a violation of the associated well-clear volume is predicted to occur along the ownship's current velocity vector within the lookahead time. A band is peripheral if it does not include the current velocity vector. Hence, a violation of the associated well-clear volume is predicted to occur within the lookahead time if the ownship maneuvers to a velocity vector included in the band. A separation assurance conflict band becomes a *recovery band* if loss of well clear has already occurred, or cannot be avoided. Recovery bands provide maneuver guidance to regain well-clear status in a timely manner within the ownship performance limits, while avoiding the violation of a smaller separation volume.

In compliance with DO-365 MOPS requirements, DAIDALUS implements an alerting scheme, which is designed to compute a numerical value indicating the severity

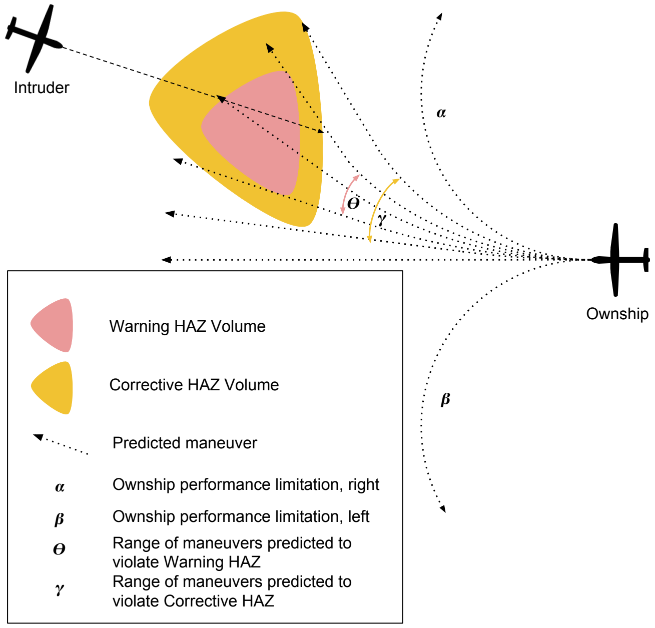
Intruder
α
θ
γ
Ownship
β
Warning HAZ Volume
Corrective HAZ Volume
Predicted maneuver
α Ownship performance limitation, right
β Ownship performance limitation, left
θ Range of maneuvers predicted to violate Warning HAZ
γ Range of maneuvers predicted to violate Corrective HAZ

Fig. 3. Separation Assurance Bands for Warning and Corrective Hazard Volumes

of a potential loss of well-clear. This numerical value is based on predicted time to violation of a series of hazard volumes that extend the standard well-clear definition. The hazard volumes are called Warning HAZ Volume, Corrective HAZ Volume, and Predictive HAZ Volume and they are used for the alerting logic, as well as for the bands computation. These volumes are defined using the same formula as the standard well-clear volume, but with parameters that make them more conservative than the standard well-clear volume. The DAIDALUS alerting scheme is thus based on the prediction of well-clear violations for different sets of increasingly conservative threshold values, within increasingly shorter alerting times. In general, the scheme yields alert levels that increase in severity as a potential pairwise conflict scenario increases in risk.

The DAIDALUS core algorithms assume a Euclidean 3-dimensional coordinate system. This coordinate system is based on a projection of the ownship and traffic geodesic coordinates into the plane that is tangent to the earth at sea level at the ownship's position. DAIDALUS uses a few key mathematical functions, which are used to define the violation volumes, alerting, and bands algorithms. These functions take as inputs position and velocity vectors for two aircraft. The horizontal (x,y-coordinates) of the ownship's position and velocity are written $\mathbf{s}_o$ and $\mathbf{v}_o$, and the intruder's are written $\mathbf{s}_i$ and $\mathbf{v}_i$, respectively. The corresponding z-coordinates are written s_{oz}, v_{oz}, s_{iz}, and v_{iz}, respectively. Most of the functions in DAIDALUS take as inputs the relative horizontal position $\mathbf{s} = \mathbf{s}_o - \mathbf{s}_i$, relative horizontal velocity $\mathbf{v} = \mathbf{v}_o - \mathbf{v}_i$, relative vertical position $s_z = s_{oz} - s_{iz}$, and relative vertical velocity $v_z = v_{oz} - v_{iz}$ of the ownship with respect to the intruder. The core functions that DAIDALUS uses are given as follows.

- Horizontal range:

$$r(t) = \|\mathbf{s} + t\mathbf{v}\| \equiv \sqrt{\mathbf{s}^2 + 2t(\mathbf{s}\cdot\mathbf{v}) + t^2\mathbf{v}^2}.$$

- Time to Horizontal Closest Point of Approach (TCPA):

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

- Horizontal Distance at TCPA:

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

- Vertical range at time t:

$$r_z(t) \equiv |s_z + tv_z|.$$

- Time to Co-Altitude:

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

- Modified Tau:

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

The well-clear function WCV returns the value true if the aircraft are in well-clear violation at the current time, and it depends on configurable parameters. By default, they are set to the following values: DMOD = HMD = 4000 ft, ZTHR = 450 ft, TAUMOD=35 s, TCOA = 0 s. It is assumed that HMD = DMOD.

$$\begin{aligned} \mathtt{WCV}(\mathbf{s},\mathbf{v},s_z,v_z) \equiv & \mathtt{Horizontal\_WCV}(\mathbf{s},\mathbf{v}) \text{ and} \\ & \mathtt{Vertical\_WCV}(s_z,v_z), \end{aligned} \tag{1}$$

where

$$\begin{aligned} & \mathtt{Horizontal\_WCV}(\mathbf{s},\mathbf{v}) \equiv \\ & \quad \|\mathbf{s}\| \leq \mathtt{DMOD} \text{ or} \\ & \quad (d_{\text{cpa}}(\mathbf{s},\mathbf{v}) \leq \mathtt{HMD} \text{ and } 0 \leq \tau_{\text{mod}}(\mathbf{s},\mathbf{v}) \leq \mathtt{TAUMOD}), \end{aligned} \tag{2}$$

and

$$\begin{aligned} & \mathtt{Vertical\_WCV}(s_z,v_z) \equiv \\ & \quad |s_z| \leq \mathtt{ZTHR} \text{ or } 0 \leq t_{\text{coa}}(s_z,v_z) \leq \mathtt{TCOA}. \end{aligned} \tag{3}$$

The detection, alerting, and bands algorithms described in [2] depend on each of the above functions.

III. Dynamic Well Clear Volumes

DAIDALUS version 1.0 used a single alerting and guidance processing scheme, called an *alerter*, for all aircraft, defaulting to one designed to meet the requirements specified by SC-228 in DO-365. Future changes to the well-clear concept are likely to include different well-clear definitions based on whether aircraft are in a terminal area, or meet other criteria such as particular size, weight and power restrictions. DAIDALUS version 2.0 addresses this need with *dynamic well clear volumes*, which allow the ownship aircraft to employ a different alerter depending on how each intruder aircraft is classified. This classification and its corresponding alerter can be based on whether the ownship, intruder, or both are in the terminal area, or it can be tailored for any new concept of operations that involves multiple well-clear definitions. In addition, the new implementation allows the alerter for a particular intruder to be updated on-the-fly in real-time.

As mentioned above, the introduction of dynamic well-clear volumes is handled in DAIDALUS through the *alerter* class. A single alerter is composed of an ordered collection of well-clear volumes and associated alerting times. These volumes are assumed to increase in severity, in the sense that the first volume should be the largest, and have the largest alert time. Each successive volume should be smaller (or equal to) the previous, and have an alerting time no larger than the previous alerting time. In this sense, being predicted to enter a particular volume within the associated alerting time represents increasing levels of risk. The alert level for an intruder is then determined by the smallest volume that is predicted to be violated within the associated alerting time. For the purposes of maneuver guidance, each entry in the alerter is also assigned a *region* chosen from the four options *none, far, mid* and *near.* The regions *far, mid* and *near* are assumed to be assigned to volumes and alerting times of increasing severity as well, while the *none* region can be assigned to any volume. A band labeled with a region other than *none* indicates that the maneuvers described by the band will lead to the violation of the volume labeled by the region, within the associated alerting time.

The main difference in DAIDALUS version 2.0 is that multiple alerters can now be defined, and can be assigned to different intruder aircraft based on some determination of its status. In fact, DAIDALUS implements two different uses of multiple alerters. DAIDALUS can be utilized in either an *intruder-centric* model, where each intruder aircraft is assigned one of the alerters, or the *ownship-centric* model, where the ownship chooses one of the alerters, and uses it for all intruder aircraft. For example, an intruder-centric use might assign one alerter to all large aircraft, and assign a second alerter with smaller volumes to slower and lighter aircraft. An ownship-centric use case might use one alerter for operations that are considered "up and away", while switching to a different alerter when the ownship enters a defined terminal area. Use of the multiple alerter capabilities of DAIDALUS is done simply, by passing an alerter index along with each aircraft state. The ownship-centric model is enabled by passing the alerter index (a positive whole number) of the desired alerter with the ownship. To use the intruder-centric model, the ownship alerter index is set to 0, and each intruder state should include the index of the alerter to be used.

Providing an alert level for an intruder is straight-forward, even with the use of multiple alerters, due to the one-to-one nature of alerting. The integration of multiple alerters into the bands algorithm for maneuver guidance provided some challenges due to its one-to-many nature. For instance, in DAIDALUS version 1.0, there is a parameter in the alerter that set the *conflict level.* This was the volume in the alerter that would trigger the computation of *recovery* bands. These bands are calculated when it is determined that no maneuver will allow the ownship to avoid entering the specified well-clear volume with some intruder, and are designed to maintain a smaller safety threshold, while returning to well-clear staus in a timely fashion. In DAIDALUS 2.0, each alerter may have a different numbered volume that is chosen to initiate recovery. To unify these, the global parameter *conflict region* is now specified. This bands region is now used to initiate recovery, when the bands specified by this region saturate, meaning that there is no way to avoid entering the volume associated with this region for some intruder aircraft.

Several other such subtle changes were required to allow the integration of dynamic well-clear volumes into the 2.0 version of DAIDALUS. These are documented and example configurations given in the documentation and release notes.

IV. New Inputs to DAIDALUS

In addition to allowing an input that determines the alerter to be used for a particular intruder aircraft, DAIDALUS version 2.0 allows for uncertainty information about the positions and velocities of the aircraft to be taken into account.

The main inputs to the DAIDALUS algorithms are the *reported* relative positions and velocities given below.

$\mathbf{s}$	reported relative horizontal position
$\mathbf{v}$	reported relative vertical velocity
s_z	reported relative vertical position
v_z	reported relative vertical velocity

In the modeling framework used in the newest version of DAIDALUS and in the rest of this paper, *actual* positions and velocities are written using similar notation but with a bar over each:

$\bar{\mathbf{s}}$	actual relative horizontal position
$\bar{\mathbf{v}}$	actual relative vertical velocity
$\bar{s_z}$	actual relative vertical position
$\bar{v_z}$	actual relative vertical velocity

A key goal in implementing sensor uncertainty mitigation in DAIDALUS is to provide algorithms whose inputs are the reported state information $\mathbf{s}$, $\mathbf{v}$, s_z, and s_z, but which compute violation information for the actual state information $\bar{\mathbf{s}}$, $\bar{\mathbf{v}}$, $\bar{s_z}$, and $\bar{v_z}$.

Sensors such as ADSB and radar often provide uncertainty information in the form of variances and covariances of the components of their positions and velocities. To accommodate this, the integration of sensor uncertainty mitigation into DAIDALUS now allows DAIDALUS to take the following quantities as inputs for each aircraft, including the ownship and each intruder aircraft.

σ_{px}	std dev of x-coord of position
σ_{py}	std dev of y-coord of position
ϕ_{pxy}	signed co-std dev between x, y coords of position
σ_{vx}	std dev of x-coord of velocity
σ_{vy}	std dev of y-coord of velocity
ϕ_{vxy}	signed co-std dev between x, y coords of velocity
σ_{pz}	std dev of z-coord of position
σ_{vz}	std dev of z-coord of velocity

A signed co-std deviation between two random variables is defined by $\phi = \text{sign}(c) \cdot \sqrt{|c|}$, where c is the associated covariance between the variables, and where $\text{sign}(c)$ is -1 if $c < 0$ and 1 if $c \geq 0$.

The language in this paper refers to an aircraft as either the ownship or an intruder. In these cases, an extra subscript o or i may be added to the inputs above to indicate whether the given uncertainty metric belongs to the ownship or an intruder. For instance, the standard deviation of the x-coordinate of the intruder's position may be written σ_{pxi}.

V. Sensor Uncertainty Mitigation Via Uncertainty Ellipses

As noted in the introduction, the new functions in DAIDALUS for sensor uncertainty mitigation are based on an explicit mathematical derivation of the impact of uncertainty on the functions defining violations and alerting. In particular, they are based on analyzing the effects of uncertainty on the functions r, t_{cpa}, d_{cpa}, r_z, t_{coa}, and (in particular) τ_{mod}, which are defined in Section II. The first such approach to sensor uncertainty mitigation is explained in this section. It is based on the idea of using the covariance matrix of the sensor uncertainty information to compute bounds on the errors in the position and velocity vectors of the two aircraft. There are six tunable parameters to the resulting algorithms. The first three tunable parameters are z-scores indicating the number of standard deviations to consider for horizontal and vertical position uncertainty and for vertical velocity uncertainty:

`h_pos_z`	# std dev's for horiz. pos. uncert.
`v_pos_z`	# std dev's for vert. pos. uncert.
`v_vel_z`	# std dev's for vert. vel. uncert.

The last three tunable parameters are two z-scores and a distance. The two z-scores indicate the number of standard deviations to consider for the horizontal velocity uncertainty. The first of the z-scores is less than or equal to the second, and the algorithm phases between the first and the second as the range between the aircraft moves from this distance (the last parameter) to zero. That is, as the range between the aircraft decreases, the algorithm considers a larger uncertainty ellipse for the horizontal component of the velocity. The three tunable parameters for horizontal velocity uncertainty are:

`h_vel_z_2`	# std dev's for horiz. vel. uncert.
`h_vel_z`	# std dev's for horiz. vel. uncert.
`h_vel_dist`	# dist. at which to start phasing

The reason that a smaller uncertainty ellipse is used for the velocity uncertainty when the aircraft are further apart is that a slight change in the size of the velocity error ellipse at those distances can produce a very large set of possible future states as those velocities are propagated in time. Thus, if the algorithms use a constant-size uncertainty ellipse for the horizontal velocity, tuning algorithm parameters for a given level of conservatism could be dominated by scenarios with large ranges. This problem is mitigated by allowing the size of the horizontal velocity uncertainty ellipse to decrease as the aircraft get further apart. Thus, it is required that `h_vel_z_2` ≤ `h_vel_z`.

The number of standard deviations to consider in the uncertainty of the horizontal velocity is given by the output of the function `weighted_z_score`, which is a linear function that that takes as input the current range between the aircraft. When $\mathit{range} >$ `h_vel_dist`, the function `weighted_z_score` is defined by `h_vel_z_2`, and when $\mathit{range} \leq$ `h_vel_dist`, it is defined by the following formula.

$$\left(1 - \frac{\mathit{range}}{\texttt{h\_vel\_dist}}\right) \cdot \texttt{h\_vel\_z}+ \\ \left(\frac{\mathit{range}}{\texttt{h\_vel\_dist}}\right) \cdot \texttt{h\_vel\_z\_2},$$

The DAIDALUS logic takes the six tunable parameters above and computes four uncertainty bounds that are then propogated through the well-clear functions. The four uncertainty bounds are as follows.

`s_err`	Max uncert. in horizontal pos.
`sz_err`	Max uncert. in vertical pos.
`v_err`	Max uncert. in horizontal vel.
`vz_err`	Max uncert. in vertical vel.

The bounds `s_err` and `sz_err` are positive distances, and `v_err` and `vz_err` are positive speeds. DAIDALUS takes input state information for an ownship aircraft and a collection of intruder aircraft. The well-clear and alerting functions are computed pairwise for the ownship and for each intruder aircraft. The values of `s_err`, `sz_err`, `v_err`, and `vz_err` are computed as follows:

$$\texttt{s\_err} = \texttt{s\_err}_o + \texttt{s\_err}_i$$
$$\texttt{sz\_err} = \texttt{sz\_err}_o + \texttt{sz\_err}_i$$
$$\texttt{v\_err} = \texttt{v\_err}_o + \texttt{v\_err}_i$$
$$\texttt{vz\_err} = \texttt{vz\_err}_o + \texttt{vz\_err}_i$$

In these formulas, each subscript indicates whether the given bound is an uncertainty value for the ownship or the intruder. For the ownship, and similarly for the intruder, the vertical errors $\texttt{sz\_err}_0$ and $\texttt{vz\_err}_o$ are computed via the following formulas.

$$\texttt{sz\_err}_o = \texttt{v\_pos\_z} \cdot \sigma_{pzo}$$
$$\texttt{vz\_err}_o = \texttt{v\_pos\_z} \cdot \sigma_{vzo}$$

Here σ_{pzo} and σ_{vzo} represent the standard deviations of the vertical components of the ownship's position and velocity, respectively, as indicated at the end of Section IV.

The computations of the horizontal error distance $\texttt{s\_err}_o$ and speed $\texttt{v\_err}_o$ for the ownship, and likewise $\texttt{s\_err}_i$ and $\texttt{v\_err}_i$ for the intruder, are slightly more complicated. They are given by the diameters of the uncertainty ellipses for the ownship's horizontal position and horizontal velocity, respectively. The distances $\texttt{s\_err}_o$ and $\texttt{v\_err}_o$ are given by the following formulas

$$\begin{aligned}\texttt{s\_err}_o &= \texttt{h\_pos\_z} \cdot \max(\sqrt{|e_{so1}|}, \sqrt{|e_{so2}|})\\ \texttt{v\_err}_o &= \texttt{weighted\_z\_score}(range) \cdot \\ &\qquad \max(\sqrt{|e_{vo1}|}, \sqrt{|e_{vo2}|})\end{aligned} \tag{4}$$

In (4), e_{so1}, e_{so2} and e_{vo1}, e_{vo2} are the eigenvalues of the covariance matrices

$$\begin{pmatrix} \sigma^2_{pxo} & \text{sign}(\phi_{pxyo}) \cdot \phi^2_{pxyo} \\ \text{sign}(\phi_{pxyo}) \cdot \phi^2_{pxyo} & \sigma^2_{pyo} \end{pmatrix} \text{ and } \begin{pmatrix} \sigma^2_{vxo} & \text{sign}(\phi_{vxyo}) \cdot \phi^2_{vxyo} \\ \text{sign}(\phi_{vxyo}) \cdot \phi^2_{vxyo} & \sigma^2_{vyo} \end{pmatrix},$$

respectively, and $range$ is the current reported distance between the aircraft.

Once the error bounds `s_err`, `sz_err`, `v_err`, and `vz_err` are computed using the formulas above, these values are propogated through the functions r, t_{cpa}, d_{cpa}, r_z, t_{coa}, and τ_{mod} (defined in Section II) to determine whether the aircraft might be in violation or whether an alert might be needed given the uncertainty levels quantified by the parameters `h_pos_z`, `v_pos_z`, `v_vel_z`, `h_vel_z_2`, `h_vel_z`, and `h_vel_dist`.

DAIDALUS defines functions

```
WCV_taumod_uncertain_at
WCV_taumod_uncertain_interval
WCV_taumod_uncertain_detection
```

with inputs **s** (reported horizontal component of relative position), **v** (reported horizontal component of relative velocity), s_z (reported relative altitude), v_z (reported relative vertical speed), `s_err`, `v_err`, `sz_err`, and `vz_err`. The first function has an extra input t, the second has an extra input T, and the third has extra inputs B and T. There are correctness theorems for each of these three functions that state that these algorithms conservatively compute violation information for the actual state information $\bar{\mathbf{s}}$, $\bar{\mathbf{v}}$, $\bar{s_z}$, and $\bar{v_z}$ (defined in Section IV), when they are given as inputs the reported state information **s**, **v**, s_z, and v_z. The correctness theorems have hypotheses that $|\bar{\mathbf{s}} - \mathbf{s}| \leq \texttt{s\_err}$, $|\bar{\mathbf{v}} - \mathbf{v}| \leq \texttt{v\_err}$, $|\bar{s_z} - s_z| \leq \texttt{sz\_err}$, and $|\bar{v_z} - v_z| \leq \texttt{vz\_err}$. The correctness theorems state that these three algorithms are *conservative* in the sense that if there is a violation for the actual positions and velocities and these hypotheses hold (the errors are within the given bounds), then the algorithms will report violations for the reported positions and velocities. Thus, these theorems state that violation information for the actual positions and velocities can be computed by executing these algorithms on the reported positions and velocities. The theorems also assume that during the relevant time windows, the positions of the aircraft will be linear projections of their current positions along their current velocities.

The function `WCV_taumod_uncertain_at` detects whether a well clear violation is possible at time t, where the well-clear volume is defined using configurable size parameters such as `TCOA`, `ZTHR`, and `TAUMOD`. The function `WCV_taumod_uncertain_interval` computes a time interval during the time window $[0, T]$ at which there may be a violation of well clear. The function `WCV_taumod_uncertain_detection` returns `True` if is possible, given the error bounds, that there is a well-clear violation during the input time interval $[B, T]$. These three functions are then used in DAIDALUS to compute bands, well-clear, and alerting information.

VI. Sensor Uncertainty Mitigation Via Probability Bounds

As mentioned earlier, DAIDALUS provides a second, alternative, approach to handling sensor uncertainty. The functions provided in this second modeling framework have a different parameter than those based on uncertainty ellipses, namely a probability P_{thresh}. The algorithms use formally proved formulas for the means and variances of random variable versions of well-clear definition functions r, t_{cpa}, d_{cpa}, r_z, t_{coa}, and τ_{mod}. They only predict a violation when these formulas imply that the probability of a violation with the actual position/velocity state information (not the reported states) may be greater than P_{thresh}. This probability is configurable and can be raised to make the algorithm less likely to alert or lowered to make it more likely.

It should be noted that while DAIDALUS now provides this second set of functions for handling sensor uncertainty, either this approach or the approach based on uncertainty ellipses (Section V) will eventually be chosen as superior, at which point the other approach may be be abandoned from operational uses of DAIDALUS.

The algorithms explicitly compute an upper bound on the number of standard deviations from the mean that are required to guarantee that a sample has a probability below a given threshold. This is accomplished through a function `cm`:

$$\texttt{cm}(p) = \sqrt{\frac{1}{p} - 1}.$$

This function, which is called a Cantelli Multiplier in DAIDALUS, is defined when p is a probability that is less than 1 and greater than 0. The correctness statement of this function is that

$$\texttt{Prob}(|X - \mu| \geq \texttt{cm}(p) \cdot \sigma) \leq 2 \cdot p$$

for *any* random variable X with mean μ and standard deviation σ, a result that follows from Cantelli's inequality.

The modeling framework for this approach assumes that there are four random variables that quantify the uncertainty in sensor readings:

X	error in x coord. of reported relative pos.
Y	error in y coord. of reported relative pos.
Pz	error in z coord. of reported relative pos.
Vz	error in x coord. of reported relative velocity.

It is assumed in this modeling framework (but not the framework based on uncertainty ellipses) that the vector representing the error in the x and y coordinates of the relative velocity is given by

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} \texttt{X} \\ \texttt{Y} \end{pmatrix},$$

where $a \cdot d - c \cdot b \neq 0$. The real numbers a, b, c, and d are explicit inputs to the algorithms associated with this modeling approach. This assumption that the horizontal relative velocity uncertainty is a matrix multiple of the horizontal relative position uncertainty is a simplifying assumption that can be viewed as a model of the correlation between the position and velocity uncertainties. The inputs to the DAIDALUS algorithms are the *reported* positions and velocities $\mathbf{s}$, $\mathbf{v}$, s_z, and v_z, as defined in Section IV. The relationships between the actual and reported positions and velocities are given by the following relationships.

$$\begin{aligned} \bar{\mathbf{s}} &= \mathbf{s} + \begin{pmatrix} X \\ Y \end{pmatrix} \\ \bar{\mathbf{v}} &= \mathbf{v} + \begin{pmatrix} a & b \\ c & d \end{pmatrix} \cdot \begin{pmatrix} X \\ Y \end{pmatrix} \\ \bar{s_z} &= s_z + \texttt{Pz} \\ \bar{v_z} &= v_z + \texttt{Vz} \end{aligned}$$

As in Section IV, $\bar{\mathbf{s}}$, $\bar{v}$, $\bar{s_z}$, and $\bar{v_z}$ define the actual relative state information for the aircraft. Note that the formulas above imply that for this modelling framework, $\bar{\mathbf{s}}$, $\bar{v}$, $\bar{s_z}$, and $\bar{v_z}$ are random variables. Thus, evaluating the condition $\texttt{WCV}(\bar{\mathbf{s}}, \bar{\mathbf{v}}, \bar{s_z}, \bar{v_z})$ on the actual positions and velocities is a boolean expression with atoms given by algebraic inequalities involving the random variables X, Y, $\texttt{Pz}$, and $\texttt{Vz}$. Thus, $\texttt{Prob}(\texttt{WCV}(\bar{\mathbf{s}}, \bar{\mathbf{v}}, \bar{s_z}, \bar{v_z}))$ is a well-defined probability. The algorithms described below compute violation information for the actual state information, and their correctness theorems give provable bounds on this probability.

Similarly to the algorithms described in Section II based on uncertainty ellipses, DAIDALUS defines functions

WCV_taumod_prob_at
WCV_taumod_prob_interval
WCV_taumod_prob_detection

with inputs $\mathbf{s}$ (reported horizontal relative position), $\mathbf{v}$ (reported horizontal relative velocity), s_z (reported relative altitude), v_z (reported relative vertical speed), P_{thresh} (a probability), and real numbers VarX, VarY, CovXY, VarPz, and VarVz. When the functions are evaluated in DAIDALUS, these last five inputs are set to the appropriate variances and covariances of X, Y, Pz, and Vz, as indicated below:

$$\begin{aligned} \texttt{VarX} &= \sigma_{pxi}^2 \\ \texttt{VarY} &= \sigma_{pyi}^2 \\ \texttt{CovXY} &= \text{sign}(\phi_{pxyi}) \cdot \phi_{pxyi}^2 \\ \texttt{VarPz} &= \sigma_{pzi}^2 \\ \texttt{VarVz} &= \sigma_{vzi}^2 \end{aligned}$$

The standard deviations σ_{px}, σ_{py}, ϕ_{pxy}, σ_{vx}, σ_{vy}, ϕ_{vxy}, σ_{pz}, and σ_{vz} are defined in Section IV.

The function WCV_taumod_prob_at has an extra input t (a time), the function WCV_taumod_prob_interval has an extra input T (also a time), and WCV_taumod_prob_detection has extra inputs B and T (also times). There are correctness theorems for each of these three functions. The correctness theorems state that these three algorithms, when evaluated on the reported state information, conservatively predict when the probability of a violation (on the actual state information) will be greater than P_{thresh}. The probability value P_{thresh} is a probability that can be raised to make the algorithm more conservative or lowered to make it less conservative. The function WCV_taumod_prob_at detects whether the probability of violation at the current state is greater than P_{thresh}. The function WCV_taumod_prob_interval computes a time interval during the time window $[0, T]$ that is guaranteed to contain every time at which the probability of violation (along a linearly projected trajectory) is greater than P_{thresh}. The function WCV_taumod_prob_detection returns True if there is a time during the interval $[B, T]$ during which the probability of violation (along a linearly projected trajectory) is greater than P_{thresh}. These three functions are then used in DAIDALUS to compute bands, well-clear, and alerting information.

VII. Future Work

As noted in the introduction, DAIDALUS is the reference implemenation of detect and avoid for unmanned aircraft systems chosen by RTCA SC-228, and it is included in its corresponding Minimum Operational Performance Standards (MOPS) document, DO-365. The dynamic well clear volumes and uncertainty mitigation algorithms presented in this paper address needs of SC-228. As such, there are a few areas of work still needed to futher refine these DAIDALUS additions to make them usable by SC-228 and other users.

One area of future work is in the area of formal verification. Specifically, the developers of DAIDALUS plan to apply an approach called *model animation* [4] [2] to validate that the code implemenations of the new algorithms faithfully represent the formally verified algorithms. This process was completed for previous versions of DAIDALUS. Model animation involves extracting output values from the algorithm implementations in both the formal specification language of PVS and the programming language. These output values are computed on a suite of test cases. The outputs of the two implementations are compared to determine if they agree to a specified precision. This process helps remove problems arising during the translation from the formal specification language into code.

Another area of future work is tuning the parameters of the algorithms for sensor uncertainty mitigation. Section V defines six tunable parameters that affect the behavior of the uncertainty-ellipse based algorithms. These are `h_pos_z`, `v_pos_z`, `v_vel_z`, `h_vel_z_2`, `h_vel_z`, and `h_vel_dist`. Increasing each of these parameters makes the resulting algorithms more conservative. Choosing the correct values for these parameters is an operational problem and requires examing the behavior of the algorithms with different parameters. Future work in the near term will focus on parameter refinement and will aim to find the right values to satisfy alert timing requirements found in the RTCA SC-228 MOPS document DO-365.

Section VI also defines algorithms based on setting a probability threshold P_{thresh} and alerting when the probability of a violation exceeds that value. Future work may also focus on setting the correct value for the parameter P_{thresh} using the same approach as described above for uncertainty ellipses.

Finally, if the approach from Section V, based on uncertainty ellipses, presents operational concerns, the approach in Section VI will be examined as a potential alternative.

VIII. Conclusion

This paper presents the following two additions to DAIDALUS, which support requirements developed by RTCA Special Committee 228 (SC-228) for unmanned aircraft in the national airspace.

- dynamic well clear volumes, and
- sensor uncertainty mitigation.

Dynamic well-clear volumes provide the capability to define distinct alerters for different well-clear definitions, and assign them to intruders as desired, including the ability to change this alerter on-the-fly. In particular, DAIDALUS implements two ways to use this capability. The ownship-centric model changes the alerter used for every intruder based on some determination by the ownship, while the intruder-centric model allows for each intruder to be assigned an alerter independently.

This paper also describes new functions in DAIDALUS for both computing and detecting well-clear violations in the presence of sensor uncertainties. These sensor uncertainty mitigation algorithms take as inputs the uncertainties in aircraft positions and velocities, typically described by variances and covariances on their east-north-up components. Future work in the near term will focus on setting parameter values in these algorithms to satisfy alert timing requirements found in the RTCA SC-228 MOPS document DO-365.

The algorithms presented in this paper have been *formally proved* in the PVS [3] theorem prover. This means that their implementations in PVS are mathematically correct in the presence of certain simplifying assumptions on the trajectories of the aircraft. They were translated by hand to code, and future work will partially address agreement between the PVS and code implementations. The algorithms for handling uncertainty are noteworthy because rather than being defined through a simulation approach, they are the result of mathematically derived formulas stating the effects of uncertainty on the well-clear functions at the core of DAIDALUS.

References

[1] M. Consiglio, J. Chamberlain, C. Muñoz, and K. Hoffler, "Concept of integration for UAS operations in the NAS," in *Proceedings of 28th International Congress of the Aeronautical Sciences, ICAS 2012*, Brisbane, Australia, 2012.

[2] C. Muñoz, A. Narkawicz, G. Hagen, J. Upchurch, A. Dutle, and M. Consiglio, "DAIDALUS: Detect and Avoid Alerting Logic for Unmanned Systems," in *Proceedings of the 34th Digital Avionics Systems Conference (DASC 2015)*, Prague, Czech Republic, September 2015.

[3] S. Owre, J. Rushby, and N. Shankar, "PVS: A prototype verification system," in *Proceeding of the 11th International Conference on Automated Deductioncade*, ser. Lecture Notes in Artificial Intelligence, D. Kapur, Ed., vol. 607. Springer, June 1992, pp. 748–752.

[4] A. Dutle, C. Muñoz, A. Narkawicz, and R. Butler, "Software validation via model animation," in *Proceedings of the 9th International Conference on Tests & Proofs (TAP 2015)*, ser. Lecture Notes in Computer Science, J. Blanchette and N. Kosmatov, Eds., vol. 9154. L'Aquila, Italy: Springer, July 2015, pp. 92–108.