DAIDALUS: DETECT AND AVOID ALERTING LOGIC FOR UNMANNED SYSTEMS

César Muñoz, Anthony Narkawicz, George Hagen, Jason Upchurch, Aaron Dutle, María Consiglio
NASA Langley Research Center, Hampton, VA
James Chamberlain
Sunrise Aviation, Inc., Newport News, VA

Abstract

This paper presents DAIDALUS (Detect and Avoid Alerting Logic for Unmanned Systems), a reference implementation of a detect and avoid concept intended to support the integration of Unmanned Aircraft Systems into civil airspace. DAIDALUS consists of self-separation and alerting algorithms that provide situational awareness to UAS remote pilots. These algorithms have been formally specified in a mathematical notation and verified for correctness in an interactive theorem prover. The software implementation has been verified against the formal models and validated against multiple stressing cases jointly developed by the US Air Force Research Laboratory, MIT Lincoln Laboratory, and NASA. The DAIDALUS reference implementation is currently under consideration for inclusion in the appendices to the Minimum Operational Performance Standards for Unmanned Aircraft Systems presently being developed by RTCA Special Committee 228.

Nomenclature

Acronyms

CAT	Collision Avoidance Threshold
DAA	Detect and Avoid
DAIDALUS	Detect and Avoid Alerting Logic for Unmanned Systems
MOPS	Minimum Operational Performance Standards
NAS	National Airspace System
NMAC	Near Mid-Air Collision
PIC	Pilot in Command
RA	Resolution Advisory
SAA	Sense and Avoid
SI	International System of Units
SST	Self-Separation Threshold
TCAS	Traffic Alert and Collision Avoidance System
UAS	Unmanned Aircraft System
WCV	Well-Clear Violation Volume

Symbol

$[B,T]$	Lookahead time interval, where $0 \leq B < T$
D	Horizontal distance
H	Vertical distance
NMAC_D	Diameter of NMAC cylinder
NMAC_H	Height of NMAC cylinder
$\mathtt{DMOD}$	Modified distance threshold
$\mathtt{TAUMOD}$	Modified tau threshold
$\mathtt{HMD}$	Horizontal Miss Distance
d_{cpa}	Distance at closest point of approach
ϵ	Numerical parameter with value ±1
$\mathbf{s},\mathbf{v}$	Two-dimensional aircraft state, horizontal position and velocity, respectively
s_z, v_z	Vertical aircraft state, altitude and vertical speed, respectively
τ_{mod}	Modified tau (time function of states)
t_{cpa}	Time to closest point of approach
t_{coa}	Time to co-altitude
$\mathtt{ZTHR}$	Vertical distance threshold
$\mathtt{TCOA}$	Vertical time threshold
$[t_{\mathrm{in}}, t_{\mathrm{out}}]$	Time interval of well-clear violation, i.e., t_{in} is time to well clear violation, and t_{out} is time to exit well-clear violation
$\mathtt{own}$	Ownship state
$\mathtt{int}$	Intruder state

Subscripts

o, i	Ownship and intruder information of a position or velocity vector
x, y, z	Northern, eastern, and altitude component of a position or velocity vector

Introduction

NASA's Unmanned Aircraft Systems Integration in the National Airspace System (UAS in the NAS) project aims to develop key capabilities to enable routine and safe access of public and civil use unmanned aircraft systems (UAS) to non-segregated airspace operations. As part of the UAS in the NAS project, NASA has developed a *detect and avoid (DAA)* concept for UAS [1] that extends the sense and avoid (SAA)[1] concept outlined in the final report of the FAA-sponsored Sense and Avoid Workshop for UAS [2], wherein *sense and avoid* is defined as "the capability of a UAS to remain well clear from and avoid collisions with other airborne traffic." In support of this capability, the NASA DAA concept includes a mathematical definition of well clear to characterize a well-clear boundary and a suite of algorithms that provide situational awareness of this well-clear boundary to UAS operators.

The well-clear boundary defines a volume, referred to as the *well-clear violation volume (WCV)*, such that aircraft pairs jointly occupying this volume are considered to be in a well-clear violation [3]. This volume is intended to be both large enough to prevent safety concerns for controllers and see-and-avoid pilots and small enough to avoid disruptions to traffic flow. Formally, the WCV is defined by a boolean predicate on the states of two aircraft, i.e., their position and velocity vectors at the current time. In particular, two aircraft are *well clear* of each other if appropriate distance and time variables, determined by the relative aircraft states, remain outside a set of predefined threshold values. These distance and time variables are closely related to variables used in the Resolution Advisory (RA) logic of the Traffic Alert and Collision Avoidance System II Version 7.1 (TCAS II) [4].

The NASA DAA concept includes a suite of algorithms called DAIDALUS (Detect and Avoid Alerting Logic for Unmanned Systems). The top-level functionality provided by DAIDALUS is situational awareness to UAS operators in the form of maneuver guidance intended to aid in:

1) maintaining well-clear status, or
2) regaining separation if a well-clear violation has already occurred or a well-clear violation is unavoidable.

DAIDALUS includes algorithms for determining the well-clear status between pairs of aircraft at the current time and for predicting a well-clear violation within a given lookahead time, assuming non-maneuvering trajectories. In the case of a predicted well-clear violation, DAIDALUS also computes the time interval of the well-clear violation. Furthermore, DAIDALUS implements algorithms for computing *conflict bands*, assuming a simple kinematic trajectory model for the ownship aircraft. These bands represent ranges of track (or heading), ground speed (or ground track), and vertical speed maneuvers that are predicted to result in well-clear violation with one of more traffic aircraft within a given lookahead time. Conflict bands are intended to provide information to the UAS remote pilot and assists the pilot in selecting trajectories that will remain well clear of other aircraft. When aircraft are not well clear, or when a well-clear violation is unavoidable, DAIDALUS computes well-clear *recovery bands*, which represent ranges of horizontal and vertical maneuvers that a remote pilot may take to regain well-clear status within the minimum possible time, while minimizing collision risk. Recovery bands are designed so that they do not conflict with RA maneuvers generated by systems such as TCAS II. Finally, DAIDALUS also implements two configurable alerting algorithms that return an integer value indicating the level of alert. The lowest possible returned alert level is zero, which indicates that no alert has been issued. Higher alert levels correspond to increased levels of threat of a well-clear violation.

DAIDALUS is currently under consideration for inclusion as the DAA reference implementation of the RTCA Special Committee 228 Minimum Operational Performance Standards (MOPS) for Unmanned Aircraft Systems. The remainder of this paper discusses the high-level architecture of DAIDALUS, its data requirements, and functional specifications. It also describes the validation and verification efforts aimed at increasing the confidence that the software correctly implements its functional requirements. The DAIDALUS software library is released under NASA's Open Source Agreement.[2] The formal

[1] The terms *sense and avoid* and *detect and avoid* are both used interchangeably in UAS literature.

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

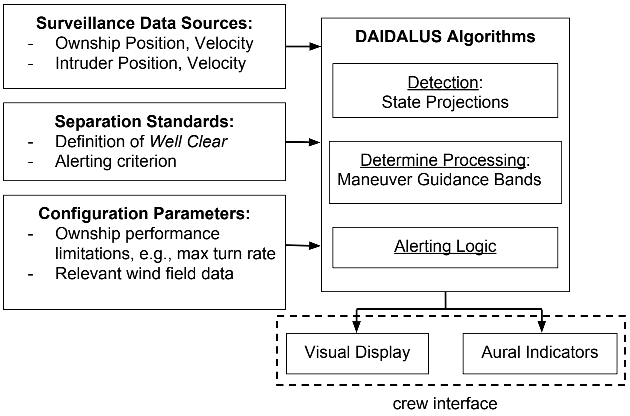
Surveillance Data Sources:
- Ownship Position, Velocity
- Intruder Position, Velocity
Separation Standards:
- Definition of Well Clear
- Alerting criterion
Configuration Parameters:
- Ownship performance limitations, e.g., max turn rate
- Relevant wind field data
DAIDALUS Algorithms
Detection: State Projections
Determine Processing: Maneuver Guidance Bands
Alerting Logic
Visual Display
Aural Indicators
crew interface

Figure 1. High-Level Architecture of DAIDALUS

models of the algorithms implemented in DAIDALUS are written in the mathematical notation of the Prototype Verification System (PVS) [5].

DAIDALUS

DAIDALUS is a software implementation intended to satisfy the operational and functional requirements detailed in NASA's DAA concept of integration for UAS [1]. The high-level functional relationship between the DAIDALUS implementation and the surveillance data sources, separation standards, and crew interface is depicted in Figure 1.

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 separation 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.

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

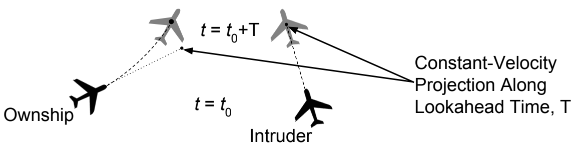

Figure 2. Constant Velocity Aircraft Projection

The detection logic computes a time interval of well-clear violation. The predictions made by the detection logic are based on pairwise, constant-velocity projections over a given lookahead time. These linear projections are illustrated in Figure 2. The own aircraft is referred to as the *ownship*, and each traffic aircraft is referred to as an *intruder*.

The maneuver guidance provided by DAIDALUS is presented in the form of *conflict bands*, i.e., ranges of ownship maneuvers that lead to a well-clear violation, or *recovery bands*, i.e., ranges of ownship maneuvers that recover from a present or unavoidable well-clear violation. The predictions used to compute these bands are based on constant turn rate and constant acceleration projections of the ownship, and constant-velocity projections of traffic aircraft. Three types of bands are provided by DAIDALUS: (1) track ranges (or heading, if wind information is provided), (2) ground speed ranges (or air speed, if wind information is provided), (3) and vertical speed ranges. As a notional example, Figure 3 illustrates state projections for the ownship and intruder aircraft used in the computation of track conflict bands.

Conflict bands may be either *preventive* or *corrective*. A band is preventive if no well-clear violation is predicted along the ownship's current velocity vector, up to the lookahead time, but some maneuver made by the ownship within its performance limitations would result in a well-clear violation within the lookahead time. A band is corrective if a well-clear violation is predicted to occur along the ownship's current velocity vector within the lookahead time. A corrective band becomes a *recovery band* if loss of well clear has already occurred, or cannot be avoided. The recovery band provides maneuver guidance to regain well-clear status in the minimum time within the ownship performance limits.

Figure 4 depicts a conceptual view of the determine-processing functionality provided by DAIDALUS, where track bands are shown for an ex-

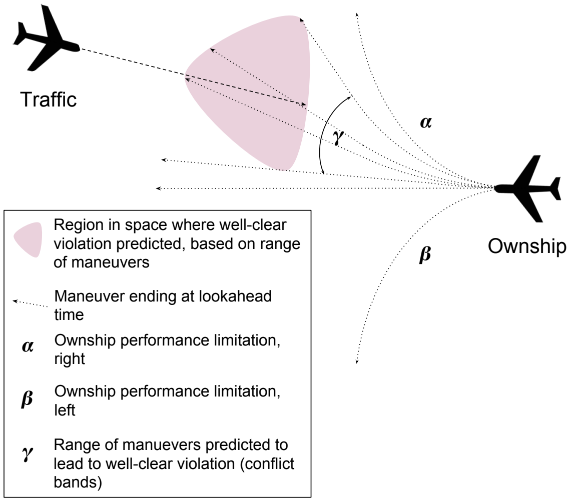
Traffic
α
γ
β
Ownship
Region in space where well-clear violation predicted, based on range of maneuvers
Maneuver ending at lookahead time
α Ownship performance limitation, right
β Ownship performance limitation, left
γ Range of manuevers predicted to lead to well-clear violation (conflict bands)

Figure 3. State Projections with Conflict Bands

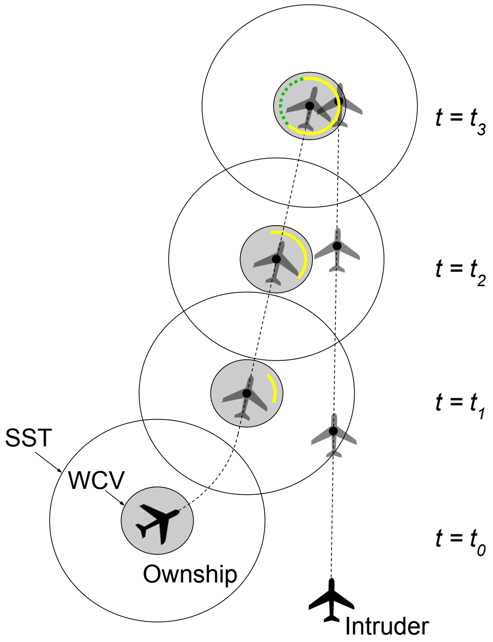

Figure 4. Maneuver Guidance Bands

ample encounter at four discrete times. The outer circle around the ownship represents the self-separation threshold (SST), which is the area relevant to the determine processing function. The inner gray area around the ownship represents the well-clear violation volume, i.e., the WCV.[3] A description of the determine processing behavior for several times of interest using the example encounter from Figure 4 follows.

- At time $t = t_0$, the ownship and intruder aircraft are depicted in their initial configuration. Since the intruder aircraft is outside the SST, no bands are displayed for the ownship.
- At time $t = t_1$, the ownship and intruder aircraft are at their new positions with the intruder inside of the SST of the ownship. Thus, the maneuver guidance calculated by DAIDALUS is presented as preventive bands (shown in amber in Figure 4), signifying the range of track maneuvers the ownship should avoid since they would lead to a well-clear violation within the lookahead time.
- At time $t = t_2$, the ownship and intruder encounter has evolved in such a way that the preventive bands from time t_1 have become corrective bands, i.e., the conflict track band has grown to include the ownship's current track. Continuing along a constant velocity is predicted to lead to a well-clear violation.
- At time $t = t_3$, the intruder is now within the WCV of the ownship, and a well-clear violation has occurred. Thus, recovery bands are now computed (shown as a dashed green arc). They provide guidance to the UAS operator as to the range of maneuvers within ownship performance limits that will allow the UAS to regain well clear in a timely manner.

DAIDALUS implements two alternative alerting schemas. One schema is based on the prediction of well-clear violations for different sets of increasingly conservative threshold values. The second schema is based on the types of bands computed for a single set of threshold values, which can be either preventive or corrective. In general, both schemas yield alert levels that increase in severity as a potential pairwise conflict scenario increases in risk. The actual alerting schema for NASA's DAA concept is still under development.

Data Requirements

Table I defines the minimum input requirements for the ownship and traffic aircraft. DAIDALUS accepts a wide set of units, including time and distance units from the International System of Units (SI).

[3] The actual shapes of the SST and WCV depend on the aircraft states.

Table I. Data input requirements

Ownship	Traffic
Identifier, e.g., call sign	Identifier, e.g., call sign
Latitude	Latitude
Longitude	Longitude
Altitude	Altitude
Ground speed	Ground speed
Airspeed (optional)	–
Ground track	Ground track
Heading (optional)	–
Vertical speed	Vertical speed

Table II. Configurable parameters

Parameter	Default value
Turn rate	3 deg/s
Bank angle	30 deg
Horizontal acceleration	2 m/s^2
Vertical acceleration	2 m/s^2
Minimum ground speed	0 knots
Maximum ground speed	700 knots
Minimum vertical speed	-5000 fpm
Maximum vertical speed	5000 fpm
Track step	1 deg
Ground speed step	1 knot
Vertical speed step	10 fpm

Northern latitudes and eastern longitudes are positive. Track and heading are provided in a clockwise from true north convention. If airspeed and heading are provided for the ownship, this information is used in conjunction with ground speed and ground track to compute local winds. The DAIDALUS algorithms then apply this wind information to aircraft current states for predicting aircraft trajectories and for computing guidance maneuvers.

The DAIDALUS bands algorithm uses the configurable parameters in Table II, whose default values are listed in the second column. The default values can be changed through the programming interface or via configuration files. Not all parameters in Table II are required. In particular, either one of turn rate or bank angle can be specified to compute track bands (heading bads, if wind information is provided). Horizontal acceleration is only used to compute ground speed bands (airspeed bands, if wind information is provided).

Table III. Data output requirements

Output	Approach
Time interval of violation	1×1
Track bands	1×N
Ground speed bands	1×N
Vertical speed bands	1×N
Alerting level	1×1

Table III shows the set of DAIDALUS outputs. The second column in the Table III indicates whether the output is computed using a 1×1 approach, i.e., pairwise, or a 1×N approach, i.e., ownship vs. traffic aircraft. Each set of bands consists of a list of intervals, representing ranges of maneuvers, and a list of elements of the enumerated type: `NONE`, `CONFLICT`, `RECOVERY`. The enumerated types `CONFLICT` and `RECOVERY` identify conflict and recovery maneuvers, respectively. The enumerated type `NONE` represents maneuvers that do not lead to well-clear violations. In the case of recovery bands, the minimum time to recover from well-clear violation, within the aircraft performance limitations, is also computed. When wind information is available, heading bands and airspeed bands are computed instead of track bands and ground speed bands, respectively.

Functional Requirements

This section describes the underlying mathematics and logic of the DAIDALUS algorithms. All of the algorithms implemented in DAIDALUS have corresponding formal specifications written in the mathematical notation of the Prototype Verification System (PVS) [5]. Furthermore, these algorithms have also been verified for functional correctness in PVS.

The algorithms presented in this section assume an 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 the Earth at sea level at the ownship's position. The following definitions are assumed. For convenience, the formulas below are presented in a relative coordinate system where the intruder aircraft is at the origin and the ownship is moving relative to the intruder, i.e., $\mathbf{s} = \mathbf{s}_o - \mathbf{s}_i$ and $\mathbf{v} = \mathbf{v}_o - \mathbf{v}_i$.

- 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:

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

Well-Clear Logic

The well-clear logic is implemented by the boolean function WCV defined in Formula (1). This function has as inputs the states of the ownship and intruder aircraft, own and int, respectively. The function returns the value true if and only if the aircraft are in well-clear violation at the current time. The threshold values are configurable parameters of the logic. By default, they are set to the following values: DMOD = HMD = 4000 ft, ZTHR = 450 ft, TAUMOD=35 s, TCOA = 0 s. Note that it is assumed that HMD = DMOD.

$$\begin{aligned} &\mathtt{WCV(own,int)} \equiv \\ &\quad \text{let } (\mathbf{s}, s_z) = \mathtt{own.pos} - \mathtt{int.pos}, \\ &\qquad (\mathbf{v}, v_z) = \mathtt{own.vel} - \mathtt{int.vel} \text{ in} \\ &\quad \mathtt{Horizontal\_WCV}(\mathbf{s},\mathbf{v}) \text{ and} \\ &\quad \mathtt{Vertical\_WCV}(s_z, v_z), \end{aligned} \tag{1}$$

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

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

Detection Logic

The well-clear detection logic is implemented by the function detection defined in Formula (4). This function has as inputs the states of the ownship and intruder aircraft, own and int, respectively, and a lookahead time interval $[B,T]$. The function returns a time interval $[t_{\text{in}}, t_{\text{out}}]$ within $[B,T]$. If $t_{\text{in}} \leq t_{\text{out}}$, the time t_{in} represents time to well-clear violation and t_{out} represents the time to exit well-clear violation, assuming constant velocity. The returned time interval is empty, i.e., $t_{\text{in}} > t_{\text{out}}$, if the aircraft are not predicted to be in violation within the interval $[B,T]$. Typically, the value of B is set to 0. However, the functions below allow for an arbitrary lookahead time interval $[B,T]$, provided that $0 \leq B < T$.

$$\begin{aligned} &\mathtt{detection}(\mathtt{own}, \mathtt{int}, B, T) : \mathbb{R}^2 \equiv \\ &\quad \text{let } (\mathbf{s}, s_z) = \mathtt{own.pos} - \mathtt{int.pos}, \\ &\qquad (\mathbf{v}, v_z) = \mathtt{own.vel} - \mathtt{int.vel} \text{ in} \\ &\quad \mathtt{det\_WCV}(\mathbf{s}, s_z, \mathbf{v}, v_z, B, T), \end{aligned} \tag{4}$$

where the function det_WCV is defined in the Appendix.

Determine-Processing Logic

The well-clear determine-processing algorithm is implemented in DAIDALUS by functions that compute maneuver guidance bands, i.e., conflict or recovery bands. When the computed conflict bands cover the full range of possible maneuvers, e.g., when the aircraft are in well-clear violation, DAIDALUS provides recovery bands. Recovery bands represent ranges of track, ground speed, and vertical speed for the ownship that lead to well-clear status in a timely manner. Computation of conflict and recovery bands is discussed subsequently.

Conflict bands are computed by incrementally projecting the maneuvers the ownship may take, within its specified performance limitations, up to the time such that a maneuver achieved. Subsequently, the ownship state is projected along a constant velocity trajectory for the the remainder of the lookahead time, i.e., the lookahead time T less the time to achieve the initial maneuver. Furthermore, each traffic aircraft is projected along a constant-velocity trajectory over the entire lookahead time. Together, these projections result in a set of maneuver guidance bands based on the most recently updated state information and are therefore influenced by, for example, sensor uncertainty and the particular update rate in use.

Recovery bands are computed by finding the smallest time, t, which is less than the lookahead time, T, such that the ownship and intruder are well clear when the aircraft states are projected to time t. In particular, these projections are made iteratively for increasing candidate values of t (less than T) over ranges of maneuvers maneuvers within the ownship's performance limits. Thus, t represents the smallest (first) time at which the ownship may escape a well-clear violation. The well-clear recovery bands algorithm guarantees that before time t, the aircraft do not violate a given minimum horizontal separation D and a given minimum vertical separation H. The values of D and H are configurable parameters. The default values of D and H are set to the DMOD and ZTHR threshold values corresponding to the TCAS II RA logic.

At the core of the functions to compute track, ground speed, and vertical speed bands is a generic, pairwise algorithm bands_1x1, defined by Formula (5), that returns a set of intervals containing maneuvers that yield well-clear violations for a given kinematic trajectory. This algorithm has as inputs:

- the relative state of the aircraft,
- a lookahead time interval, $[B,T]$,
- a current maneuver value for the ownship, c,
- respective minimum and maximum values, $u_{\min}$ and $u_{\max}$, for the maneuvers,
- a maneuver step, e,
- an acceleration, a, for the maneuver,
- horizontal and vertical distances D and H, and
- respective position and velocity functions, $\bar{p}$ and $\bar{v}$, which kinematically project the relative state of the aircraft for a given time using a constant acceleration.

Typically, the value of B is 0. When this value is non-zero, the algorithm bands_1x1 also includes ranges of values that violate the minimum separation, given by D and H, before the time T. This configuration is useful when computing recovery maneuvers.

$$\begin{aligned}&\texttt{bands\_1x1}(\mathbf{s},\mathbf{v},s_z,v_z,B,T,c,u_{\min},u_{\max},e,\\&\quad a,D,H,\bar{p},\bar{v}) : \mathrm{set}[\mathbb{R}^2] \equiv\\&\qquad \texttt{left\_1x1}(\ldots) \cup \texttt{right\_1x1}(\ldots).\end{aligned} \tag{5}$$

The auxiliary functions left_1x1 and right_1x1 are defined in the Appendix. These functions have the same parameters as bands.

From bands_1x1, the generic algorithm bands, defined by Formula (6), computes maneuver bands for an ownship and a set of traffic aircraft. In this function, the parameters own and traf represent the state of the ownship and a set of states for all traffic aircraft, respectively.

```
bands(own, traf, $B, T, c, u_{\min}, u_{\max}, e, a,$
  $D, H, \bar{p}, \bar{v}$) : $\mathrm{set}[\mathbb{R}^2] \equiv$
    $\beta := \emptyset$;
    foreach int in traf do
      let $(\mathbf{s}, s_z)$ = own.pos − int.pos,
          $(\mathbf{v}, v_z)$ = own.vel − int.vel in                (6)
      $\beta := \beta \cup$ bands_1x1($\mathbf{s}, \mathbf{v}, s_z, v_z, B, T,$
                $c, u_{\min}, u_{\max}, e, a, D, H, \bar{p}, \bar{v}$);
    endforeach
    return $\beta$,
```

where ':=' denotes the assignment operator, and ';' denotes the sequential operator for imperative pseudocode.

The maneuver ranges computed by the algorithm bands correspond to intervals of type CONFLICT. Intervals of type NONE are computed as the complement of the union of these intervals with respect to $[u_{\min}, u_{\max}]$. The algorithms that compute track, ground speed, and vertical speed for a given lookahead time, T, are defined using the following functions.

$$\begin{aligned}&\texttt{trk\_bands}(\texttt{own},\texttt{traf},T) : \mathrm{set}[\mathbb{R}^2] \equiv\\&\quad \texttt{bands}(\texttt{own},\texttt{traf},0,T,\texttt{trk}(\texttt{own}),-\pi,\pi,\\&\qquad \texttt{TrkStep},\texttt{TurnRate},0,0,\texttt{TrkPos},\texttt{TrkVel}),\end{aligned} \tag{7}$$

$$\begin{aligned}&\mathtt{gs\_bands}(\mathtt{own},\mathtt{traf},T):\mathrm{set}[\mathbb{R}^2]\equiv\\&\quad\mathtt{bands}(\mathtt{own},\mathtt{traf},0,T,\mathtt{gs}(\mathtt{own}),\mathtt{MinGs},\\&\qquad\mathtt{MaxGs},\mathtt{GsStep},\mathtt{HAccel},0,0,\mathtt{GsPos},\mathtt{GsVel}),\end{aligned}\tag{8}$$

and

$$\begin{aligned}&\mathtt{vs\_bands}(\mathtt{own},\mathtt{traf},T):\mathrm{set}[\mathbb{R}^2]\equiv\\&\quad\mathtt{bands}(\mathtt{own},\mathtt{traf},0,T,\mathtt{vs}(\mathtt{own}),\mathtt{MinVs},\\&\qquad\mathtt{MaxVs},\mathtt{VsStep},\mathtt{VAccel},0,0,\mathtt{VsPos},\mathtt{VsVel}),\end{aligned}\tag{9}$$

where

- `trk`, `gs`, and `vs` are functions that compute the current track, ground speed, and vertical speed of an aircraft, respectively;
- `MinGs` and `MaxGs` are the minimum and maximum ground speed, respectively;
- `MinVs` and `MaxVs` are the minimum and maximum vertical speed, respectively;
- `TrkStep`, `GsStep`, `VsStep` are the track, ground speed, and vertical speed steps, respectively;
- `TrkPos`,`TrkVel` are functions that kinematically project the relative position and velocity of the aircraft for a given time and constant turn rate;
- `GsPos`, `GsVel` are functions that kinematically project the relative position and velocity of the aircraft for a given time and constant horizontal acceleration;
- `VsPos`, `VsVel` are functions that kinematically project the relative position and velocity of the aircraft for a given time and constant vertical acceleration.

Recovery bands are computed using the algorithm `rec_bands`, defined by Formula (10), which is based on the generic algorithm `bands`. The algorithm `rec_bands` has as inputs:

- the state of the ownship, `own`,
- a set of states for all traffic aircraft, `traf`,
- a lookahead time, T,
- a current maneuver value for the ownship, c,
- respective minimum and maximum values for the maneuvers, $u_{\min}$ and $u_{\max}$,
- a maneuver step, e,
- an acceleration for the maneuver, a
- respective horizontal and vertical distances, D and H, and
- respective position and velocity functions, $\bar{p}$ and $\bar{v}$, which kinematically project the relative state of the aircraft for a given time using a constant acceleration.

The algorithm `rec_bands` returns a set of intervals and a time.

$$\begin{aligned}&\mathtt{rec\_bands}(\mathtt{own},\mathtt{traf},T,c,u_{\min},u_{\max},e,a,\\&\quad D,H,\bar{p},\bar{v}):[\mathrm{set}[\mathbb{R}^2],\mathbb{R}]\equiv\\&\quad\text{let }\beta=\mathtt{bands}(\mathtt{own},\mathtt{traf},0,T,c,\\&\qquad u_{\min},u_{\max},e,a,D,H,\bar{p},\bar{v})\text{ in}\\&\quad\text{if }[u_{\min},u_{\max}]\subseteq\beta\text{ then}\\&\qquad\text{let }t=\min_{0<B\le T}\{[u_{\min},u_{\max}]\not\subseteq\\&\qquad\quad\mathtt{bands}(\mathtt{own},\mathtt{traf},B,T,c,\\&\qquad\qquad u_{\min},u_{\max},e,a,D,H,\bar{p},\bar{v})\}\text{ in}\\&\qquad\text{if }t=T\text{ then }([u_{\min},u_{\max}],-1)\\&\qquad\text{else }[\mathtt{bands}(\mathtt{own},\mathtt{traf},t,T,c,\\&\qquad\quad u_{\min},u_{\max},e,a,D,H,\bar{p},\bar{v}),t]\\&\qquad\text{endif}\\&\quad\text{else}\\&\qquad(\beta,0)\\&\quad\text{endif}\end{aligned}\tag{10}$$

The maneuver ranges computed by the algorithm `rec_bands` correspond to intervals of type `CONFLICT`. Intervals of type `NONE` and `RECOVERY` are computed as follows. If the time computed by `rec_bands` is zero, the complement of the union of intervals computed by `rec_bands`, with respect to $[u_{\min},u_{\max}]$, corresponds to intervals of type `NONE`. If the time computed by `rec_bands` is negative, the whole interval $[u_{\min},u_{\max}]$ is of type `CONFLICT` and no recovery is possible within the performance limits of the aircraft for the given lookahead time T. If the time computed by `rec_bands` is greater than zero, the complement of the union of the intervals computed by `rec_bands`, with respect to $[u_{\min},u_{\max}]$, corresponds to bands of type `RECOVERY`. For a band of type `RECOVERY`, well-clear status will be recovered at the time computed by the algorithm. The recovery maneuvers are guaranteed to satisfy a given minimum horizontal separation, D, and vertical separation, H.

The algorithms that compute track, ground speed, and vertical speed for a given lookahead time, T and respective minimum horizontal and vertical separation, D and H, are defined using the following functions.

$$\begin{aligned}
&\texttt{rec\_trk\_bands}(\texttt{own}, \texttt{traf}, T, D, H) : \\
&\quad [\text{set}[\mathbb{R}^2], \mathbb{R}] \equiv \\
&\quad \texttt{rec\_bands}(\texttt{own}, \texttt{traf}, 0, T, \texttt{trk}(\texttt{own}), -\pi, \pi, \\
&\qquad \texttt{TrkStep}, \texttt{TurnRate}, D, H, \\
&\qquad \texttt{TrkPos}, \texttt{TrkVel}),
\end{aligned} \tag{11}$$

$$\begin{aligned}
&\texttt{rec\_gs\_bands}(\texttt{own}, \texttt{traf}, T) : \\
&\quad [\text{set}[\mathbb{R}^2], \mathbb{R}] \equiv \\
&\quad \texttt{rec\_bands}(\texttt{own}, \texttt{traf}, 0, T, \texttt{gs}(\texttt{own}), \texttt{MinGs}, \\
&\qquad \texttt{MaxGs}, \texttt{GsStep}, \texttt{HAccel}, D, H, \\
&\qquad \texttt{GsPos}, \texttt{GsVel}),
\end{aligned} \tag{12}$$

and

$$\begin{aligned}
&\texttt{rec\_vs\_bands}(\texttt{own}, \texttt{traf}, T) \\
&\quad [\text{set}[\mathbb{R}^2], \mathbb{R}] \equiv \\
&\quad \texttt{rec\_bands}(\texttt{own}, \texttt{traf}, 0, T, \texttt{vs}(\texttt{own}), \texttt{MinVs}, \\
&\qquad \texttt{MaxVs}, \texttt{VsStep}, \texttt{VAccel}, D, H, \\
&\qquad \texttt{VsPos}, \texttt{VsVel}).
\end{aligned} \tag{13}$$

Alerting Logic

An alerting logic provides an indication of the severity of the proximity of a particular traffic aircraft to the ownship. This indication is given as a numerical value between zero, representing no severity, and a value, k, representing the maximally-severe alert level. The greater the numerical value, the greater the severity level. DAIDALUS implements two particular alerting functions, `thresholds_alerting` and `bands_alerting`, respectively referred to as *thresholds-based alerting* and *bands-based alerting*. Both functions have as inputs the states of the ownship and the intruder aircraft. A high-level description of these functions follows, while the detailed specifications are presently under development.

The function `thresholds_alerting` also has as input a list of threshold values and alerting times denoted as $\{\texttt{DMOD}, \texttt{HMD}, \texttt{ZTHR}, \texttt{TAUMOD}, \texttt{TCOA}, T\}_{1 \leq i \leq k}$. It is assumed that the i-th threshold values are greater than or equal to the $(i+1)$-th threshold values. The function `thresholds_alerting` returns the first index, i, in the list such that $\texttt{detection}(\texttt{own}, \texttt{int}, 0, T_i)$ returns `true` for $\texttt{DMOD}_i$, $\texttt{HMD}_i$, $\texttt{ZTHR}_i$, $\texttt{TAUMOD}_i$, and $\texttt{TCOA}_i$. The function returns zero if no such index exists.

The most severe type of alert for the function `bands_alerting` corresponds to $k = 4$. Furthermore, the alerting logic implemented by this function uses only one set of threshold values: those values used to define the well-clear violation volume. The value returned by `bands_alerting` depends on the particular type of maneuver guidance computed by the DAIDALUS determine processing logic. Consequently, there is a correspondence in severity between the computed alert type and the maneuver guidance presented to the pilot. Finally, the function implements the following prioritized list of conditions.

1) If no bands are computed by `bands` for a configurable *alerting time* parameter then it returns 0.
2) If time to violation is less than a configurable time parameter then it returns 4.
3) If time to violation is less than alerting time then it returns 3.
4) If preventive bands are computed within configurable thresholds then it returns 2.
5) In any other case, it returns 1.

Verification and Validation

The DAIDALUS software was verified using the approach called *model animation* [6]. In general, the approach involves first creating formal models of the algorithms to be verified, where the properties that the algorithms are intended to possess are formally proven to hold. Secondly, the formal models are translated into the target programming language of the implementation. Finally, both the formal models and their implementations are each executed on a suite of test cases, and the outputs are compared to determine if they agree to a specified precision.

The formal models of the algorithms used in the DAIDALUS are specified in PVS, which is both a specification language and interactive theorem prover. Throughout the theorem proving functionality of the PVS, many properties of the algorithms are proven as theorems. Examples of such theorems are correctness properties of the algorithms, as well as several statements from the DAA functional requirements. For instance, there is a theorem concerning the track bands algorithm that states that for any track step maneuver, the corresponding interval is `CONFLICT` exactly when performing the maneuver would cause a loss of well-clear with another aircraft before the lookahead time. These formal proofs give a high level of assurance that the formal models of the algorithms are correct.

The reference implementations of DAIDALUS, written in Java and C++, closely mirror the formal models while taking advantage of the more expressive nature of these programming languages. In order to assure that the implementations conform to the formal models, both the formal models and their implementations were both tested on a set of 95 aircraft encounter scenarios. These scenarios were developed jointly by the USAF, Lincoln Laboratory, and NASA as a suite of stressing cases for assessing the DAA functionality.

For some of the functionality provided by DAIDALUS, testing whether the formal models and their reference implementations agree or not is straightforward. As an example, the well-clear violation logic computes a boolean value indicating the well-clear status between the ownship and a traffic aircraft. Testing if this logic is correctly implemented in software amounts to check if the same boolean value is computed in both the formal model and the software implementation. On the other hand, for algorithms that compute a numerical value the verification is more complicated since the outputs, although logically equivalent, may be different due to numerical approximations. To mitigate this, a Monte Carlo approach was used to validate the bands algorithms. After computing bands using the software implementations of the algorithms and their formal models, 400 random sample points in each band were chosen, and each point was tested to determine if the computed regions agree between the implementations. If more than 1% of the points fail to agree, then the bands were determined to be different.

The implementations were tested on the 95 scenarios, each with on average 180 time steps for testing. Each scenario was tested for computing conflict bands only, and with recovery bands. Overall, less than 0.1% of the examined steps were considered to disagree, giving a high level of confidence that the implementations match the formal models, and hence perform as desired.

Conclusion

DAIDALUS is a reference implementation of NASA's detect and avoid concept for the integration of UAS into the NAS. The underlying core logic of DAIDALUS consists of: (1) a mathematical definition of a well-clear boundary that resides inside a self-separation volume, (2) algorithms for determining if aircraft pairs have violated this well-clear boundary or are predicted to violate this boundary within a given lookahead time, and (3) a determine-processing functionality that provides both maneuver guidance to remote pilots and an alerting logic that provides an indication of severity of the proximity of traffic aircraft to the ownship. The algorithms implemented in DAIDALUS have been formally specified and verified for correctness in PVS. The software implementations have been validated using 95 stressing scenarios jointly developed by US Air Force Research Laboratory, MIT Lincoln Laboratory, and NASA. The DAIDALUS reference implementation is under consideration for inclusion as reference implementation in Minimum Operational Performance Standards for UAS developed by RTCA Special Committee 228.

DAIDALUS has been integrated into NASA's Multi Aircraft Control System (MACS)[4], a software environment for rapid prototyping of air traffic concepts. This software integration is currently used in human-in-the-loop experiments at NASA Langley Research Center to assess controller and pilot acceptability of NASA's DAA concept for UAS [7].

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] FAA Sponsored Sense and Avoid Workshop, *Sense and avoid (SAA) for unmanned aircraft systems (UAS)*, 2009.

[3] C. Muñoz, A. Narkawicz, J. Chamberlain, M. Consiglio, and J. Upchurch, "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, 2014.

[4] http://hsi.arc.nasa.gov/groups/aol/technologies/macs.php.

[4] RTCA SC-147, *RTCA-DO-185B, minimum operational performance standards for traffic alert and collision avoidance system II (TCAS II)*, 2009.

[5] S. Owre, J. Rushby, and N. Shankar, “Pvs: A prototype verification system,” in *Proceeding of the 11th International Conference on Automated Deduction-cade*, D. Kapur, Ed., ser. Lecture Notes in Artificial Intelligence, vol. 607, Springer, 1992, pp. 748–752.

[6] 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)*, J. Blanchette and N. Kosmatov, Eds., ser. Lecture Notes in Computer Science, vol. 9154, L’Aquila, Italy: Springer, 2015, pp. 92–108. DOI: 10.1007/978-3-319-21215-9_6.

[7] J. P. Chamberlain, M. C. Consiglio, J. R. C. Jr., R. W. Ghatas, and C. A. Muñoz, “NASA Controller Acceptability Study 1 (CAS-1) experiment description and initial observations,” NASA, Langley Research Center, Hampton VA 23681-2199, USA, Technical Memorandum NASA/TM-2015-218763, 2015.

34th Digital Avionics Systems Conference
September 13–17, 2015

Appendix

```
$\mathtt{det\_WCV}(\mathbf{s}, s_z, \mathbf{v}, v_z, B, T) : \mathbb{R}^2 \equiv$
  let $[t_1, t_2] = \mathtt{det\_VWCV}(s_z, v_z, B, T)$ in
    if $t_1 > t_2$ then $[T, B]$
    elseif $t_1 = t_2$ and
      $\mathtt{Horizontal\_WCV}(\mathbf{s} + t_1\mathbf{v}, \mathbf{v})$ then
        $[t_1, t_1]$                                          (14)
    elseif $t_1 = t_2$ then $[T, B]$
    else let $[t_{\text{in}}, t_{\text{out}}] =$
        $\mathtt{det\_HWCV}(\mathbf{s} + t_1\mathbf{v}, \mathbf{v}, t_2 - t_1)$ in
        $[t_{\text{in}} + t_1, t_{\text{out}} + t_1$
    endif.
```

```
$\mathtt{det\_VWCV}(s_z, v_z, B, T) : \mathbb{R}^2 \equiv$
  if $v_z = 0$ and $|s_z| \leq \mathtt{ZTHR}$ then $[B, T]$
  elseif $v_z = 0$ then $[T, B]$
  else let $[t_1, t_2] =$
    $\mathtt{vertical\_entry\_exit}(s_z, v_z)$ in              (15)
    if $T < t_1$ or $t_2 < B$ then $[T, B]$
    else $[\max(B, t_1), \min(T, t_2)]$
    endif
  endif.
```

```
$\mathtt{vertical\_entry\_exit}(s_z, v_z) : \mathbb{R}^2 \equiv$
  let $H = \max(\mathtt{ZTHR}, \mathtt{TCOA}|v_z|)$ in             (16)
    $\left[ \frac{-\mathrm{sign} v_z H - s_z}{v_z}, \frac{\mathrm{sign} v_z \mathtt{ZTHR} - s_z}{v_z} \right].$
```

```
$\mathtt{det\_HWCV}(\mathbf{s}, \mathbf{v}, T) : \mathbb{R}^2 \equiv$
  let $a = \mathbf{v}^2$,
    $b = 2(\mathbf{s} \cdot \mathbf{v}) + \mathtt{TAUMOD} \cdot \mathbf{v}^2$,
    $c = \mathbf{s}^2 + \mathtt{TAUMOD} \cdot (\mathbf{s} \cdot \mathbf{v}) - \mathtt{DMOD}^2$ in
  if $a = 0$ and $||\mathbf{s}|| \leq \mathtt{DMOD}$ then $[0, T]$
  elseif $||\mathbf{s}|| \leq \mathtt{DMOD}$ then $[0, \min(T, \Theta(\mathbf{s}, \mathbf{v}, 1))]$
  elseif $\mathbf{s} \cdot \mathbf{v} \geq 0$ or $b^2 - 4ac < 0$ then $[T, 0]$      (17)
  else let $t = \frac{-b - \sqrt{b^2 - 4ac}}{2a}$ in
    if $\Delta(\mathbf{s}, \mathbf{v}) \geq 0$ and $t \leq T$ then
      $[\max(0, t), \min(T, \Theta(\mathbf{s}, \mathbf{v}, 1))]$
    else $[T, 0]$
    endif
  endif.
```

$$\Theta(\mathbf{s}, \mathbf{v}, D, \epsilon) \equiv \frac{-\mathbf{s} \cdot \mathbf{v} + \epsilon\sqrt{\Delta(\mathbf{s}, \mathbf{v}, D)}}{\mathbf{v}^2}. \tag{18}$$

$$\Delta(\mathbf{s}, \mathbf{v}, D) \equiv D^2\mathbf{v}^2 - (\mathbf{s} \cdot \mathbf{v}^{\perp})^2. \tag{19}$$

```
left_1x1($\mathbf{s}, \mathbf{v}, s_z, v_z, B, T, c, u_{\min}, u_{\max}, e,$
  $a, D, H, \bar{p}, \bar{v}$) : set[$\mathbb{R}^2$] ≡
    let $t_e = \frac{e}{a}$ in
      $t := 0$;
      $u := c$;
      $\beta := \emptyset$;
      while $u < u_{\max}$ and $t < T$ do
          $(\mathbf{s}_t, s_{z\,t}) = \bar{p}(\mathbf{s}, \mathbf{v}, a, t)$;
          $(\mathbf{v}_t, v_{z\,t}) = \bar{v}(\mathbf{s}, \mathbf{v}, a, t)$;
          if $t < B$ then
              if $\|\mathbf{s}_t\| < D$ and $|s_{z\,t}| < H$ then
                  $\beta := \beta \cup \{[u, u_{\max}]\}$;                    (20)
                  $u := u_{\max}$;
              endif
          elseif WCV($\mathbf{s}_t, s_{z\,t}, \mathbf{v}_t, v_{z\,t}$) then
              $\beta := \beta \cup \{[u, u_{\max}]\}$;
              $u := u_{\max}$;
          elseif WCV($\mathbf{s}_t, s_{z\,t}, \mathbf{v}_t, v_{z\,t}, 0, T - t$) then
              $\beta := \beta \cup \{[u, u + e]\}$;
              $u := u + e$;
          endif
          $t := t + t_e$;
      endwhile
      return $\beta$;
```

```
right_1x1($\mathbf{s}, \mathbf{v}, s_z, v_z, B, T, c, u_{\min}, u_{\max}, e,$
  $a, D, H, \bar{p}, \bar{v}$) : set[$\mathbb{R}^2$] ≡
    let $t_e = \frac{e}{a}$ in
      $t := 0$;
      $u := c$;
      $\beta := \emptyset$;
      while$u > u_{\min}$ and $t < T$ do
          $(\mathbf{s}_t, s_{z\,t}) = \bar{p}(\mathbf{s}, \mathbf{v}, -a, t)$;
          $(\mathbf{v}_t, v_{z\,t}) = \bar{v}(\mathbf{s}, \mathbf{v}, -a, t)$;
          if $t < B$ then
              if $\|\mathbf{s}_t\| < D$ and $|s_{z\,t}| < H$ then
                  $\beta := \beta \cup \{[u_{\min}, u]\}$;                    (21)
                  $u := u_{\min}$;
              endif
          elseif WCV($\mathbf{s}_t, s_{z\,t}, \mathbf{v}_t, v_{z\,t}$) then
              $\beta := \beta \cup \{[u_{\min}, u]\}$
              $u := u_{\min}$;
          elseif WCV($\mathbf{s}_t, s_{z\,t}, \mathbf{v}_t, v_{z\,t}, 0, T - t$) then
              $\beta := \beta \cup \{[u - e, u]\}$
              $u := u - e$;
          endif
          $t := t + t_e$;
      endwhile
      return $\beta$;
```