Encounter-Based Simulation Architecture for Detect-And-Avoid Modeling

Mohamad Refai [*] and Michael Abramson[†] and Seungman Lee[‡]
Crown Consulting, Inc., Moffett Field, California, 94035

Gilbert Wu [§]
NASA Ames Research Center, Moffett Field, California, 94035

This paper presents an encounter-based simulation architecture developed to facilitate flexible and efficient detect and avoid modeling in parametric or trade-space studies on large data sets. The basic premise of this tool is that large-scale input data can be reduced to a set of "canonical encounters" and that using the reduced data in simulations does not lead to loss of fidelity. A canonical encounter is specified as ownship and intruder flight portions potentially resulting in a loss of well clear along with a set of properties that characterize the encounter. The advantages of using canonical encounters include faster simulations, reduced memory footprint, ability to select encounters based on user-specified criteria, shared encounters across multiple teams, peer-reviewed encounters, and a better understanding of the input data set, to name a few. The performance of the encounter-based approach is compared to the approach used previously, which modeled flights from departure to destination using the Java Architecture for DAA Extensibility and Modeling (JADEM). The new architecture reduced the amount of data to be processed a hundredfold and the total computation time five-fold.

Glossary

CPA	Closest Point of Approach
DAA	Detect-and-Avoid
DWC	DAA Well Clear
HITL	Human-in-the-Loop
HMD	Horizontal Miss Distance
JADEM	Java Architecture for DAA Extensibility and Modeling
Low SWaP	Low Size, Weight, and Power
LoDWC	Loss of DAA Well Clear
MOPS	Minimum Operational Performance Standards
NAS	National Airspace System
NMAC	Near Mid-Air Collision
RADES	84th Radar Evaluation Squadron
UAS	Unmanned Aircraft Systems
VFR	Visual Flight Rules
VMD	Vertical Miss Distance

I. Introduction

SUCCESSFUL integration of Unmanned Aircraft Systems (UAS) into the National Airspace System (NAS) is predicated upon maintaining or exceeding the level of safety and performance achieved by current operations [1]. One of the key

[*]Senior Engineer, M/S 210-8, mohamad.s.refai@nasa.gov
[†]Senior Engineer, M/S 210-8, AIAA Member, michael.abramson@nasa.gov
[‡]Senior Scientist, M/S 210-8, AIAA Member, seungman.lee@nasa.gov
[§]Aerospace Engineer, Aviation Systems Division, M/S 210-10, AIAA Member, gilbert.wu@nasa.gov

integration challenges is Detect and Avoid (DAA) capability, which is required for maintaining safe separation between UAS and other aircraft. To ensure that DAA safety and performance goals are achievable, various data collection and analysis methods are employed in developing UAS Minimum Operational Performance Standards (MOPS) [2]. These methods include first principle engineering analyses, human-in-the-loop simulations, flight tests, and fast-time simulations. Regardless of the methodologies employed, a main challenge is identifying representative data sets to use in foundational studies and safety and performance assessments. For National Airspace System-wide (NAS-wide) assessments, the challenge is two-fold. First, UAS mission data are not available at the anticipated densities. Second, UAS will operate in controlled and uncontrolled airspace, so, Visual Flight Rules (VFR) traffic data are also required in uncontrolled airspace.

An approach to provide representative data sets was developed by MIT Lincoln Laboratory [3], wherein radar data from the 84th Radar Evaluation Squadron (RADES) were used to create a database of statistical features of flights. These statistics are then sampled to create new trajectories with characteristics consistent with the original trajectories. Encounters are modeled by sampling one or more trajectories that satisfy "proximity" criteria. The model therefore generates encounters representing VFR to VFR encounters. This approach is henceforth referred to as the "MIT encounter model".

The model was used in [4] to investigate well clear boundaries in space and time as a function of risk of Near Mid-Air Collisions (NMAC) and Traffic Collision Avoidance System Resolution Advisory (TCAS RA) issuance. In [5] the model was used to validate an analytic approach for mapping surveillance errors to collision risk. In [6] the encounter model was used in stress testing the Airborne Collision Avoidance System (ACAS X) using a reinforcement learning algorithm designed to maximize the likelihood of finding encounters that result in a rare critical event such as an NMAC.

Another approach applies parametric variations on encounter variables to assess the performance of DAA concepts over the domain of coverage. Some of the encounter variables include range, horizontal and vertical speed, angle of approach, and turn rate. This method was used in [7], for example, to explore maneuver initiation range and other characteristics of a DAA Well Clear (DWC) definition. The DWC investigated is the modified tau definition, which has since been adopted for the MOPS [2]. Note that this approach does not take into account the probability or likelihood that a UAS will experience a particular encounter in the set of encounters thus obtained.

A third approach taken by NASA has been to develop a set of UAS flight plans that represent today's view of future predicted UAS operations and missions [8]. Nineteen missions and their corresponding geographical locations and operating schedules were developed with extensive contributions from subject matter experts [9]. To model intruder traffic, RADES data for 21 days of varying traffic density were selected and processed to provide smoothed VFR trajectories [10]. For the remainder of this section, this approach is referred to as the "NASA encounter model". It has been used to conduct several foundational NAS-wide and parametric fast-time simulations in support of MOPS development [8, 11–15].

Several studies have used multiple approaches to evaluate performance of DAA concepts. In [16] both NASA and MIT encounter models were used to evaluate three candidate well clear definitions and tune their parameters. The NASA encounter model was used, for example, to compute unmitigated rates per flight hour for TCAS RA and well clear violations. The MIT encounter model was used to provide mitigated and unmitigated probabilities and distributions of various events such as TCAS RA and well clear violations. In [17] the NASA encounter model was used to evaluate several candidate DWC definitions for low cost, size, weight, and power UAS. The MIT encounter model was used to complement and validate results of the NASA encounter model.

While the NASA encounter model has proven quite valuable to MOPS development, it suffers from several practical shortcomings, which include:

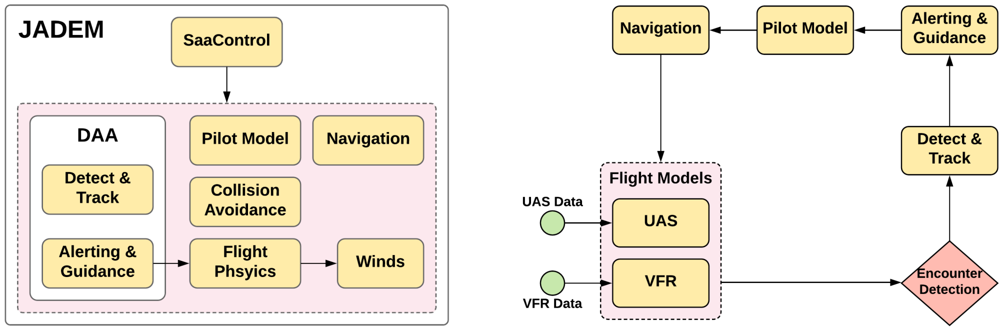
- The total duration of encounters is a small fraction of the time of flight; this makes a full end-to-end simulation inefficient.
- UAS maneuvers, when employed, alter the remaining UAS trajectory and the characteristics of following encounters, making data comparison difficult.
- VFR data are rather extensive and consequently, simulations consume considerable resources, despite various optimizations.
- Results include considerably more data than are typically required.
- A priori scenario selection is not possible.
- The data are not currently shared mainly due to size and computational requirements.
- Simulation results can be moderately sensitive to flight mechanics models, computational optimizations, etc; this complicates comparisons of alternative approaches.

This paper describes an encounter-based simulation architecture aimed at addressing these challenges. Simply put, this new approach uses the RADES and UAS mission data to identify and persist potential encounters, which are subsequently used in various Detect and Avoid (DAA) simulations and analyses. The process of identifying encounters also generates a set of corresponding encounter properties, which can be used to pre-select encounters of interest to a particular study. This approach marks a shift from past NAS-wide simulations by focusing on the events of interest and simulating only the relevant portions of flight instead of full end-to-end departure to destination modeling. This allows efficient processing of realistic encounters that are true to the original data. In addition, the new architecture enables flexible processing by allowing processing steps to be omitted, interspersed, or their order changed to suit research needs.

In what follows, we discuss this approach and its benefits in more detail. The next section provides background information on past simulation capability and its limitations. Section III defines encounters and scenarios and describes the encounter-based simulation architecture and how it addresses the existing limitations. Section V describes encounter detection. Section VI discusses encounter properties and their uses. Section VII describes use of encounter properties for filtering and scenario selection and provides typical use cases. Performance evaluation results for the new architecture are summarized in Section VIII. Finally, concluding remarks are presented in Section IX.

II. Background

The Java Architecture for DAA Extensibility and Modeling (JADEM) [18] was designed as a general purpose simulation tool for evaluating DAA concepts and their safety characteristics. As such, it was built to support testing, NAS-wide assessments, and parametric trade-space studies. JADEM is composed of several models (see Fig. 1a), which include Flight Physics, surveillance Detect and Track, Alerting and Guidance, Collision Avoidance, Pilot, Navigation, and Wind Models. The models are managed by a driver called SaaControl, which provides the required simulation functionality.



(a) JADEM Models **(b) JADEM Functional Schematic**

Fig. 1 JADEM

Figure 1b depicts a functional schematic of a typical SaaControl simulation. VFR and UAS flight mission data are ingested periodically and modeled to provide discrete truth states. The states are processed in time windows, typically five minutes. For each time window, "Encounter Detection" uses proximity of aircraft to determine whether an encounter is possible and creates all possible encounters over the time window. Encounter truth states are processed through the surveillance "Detect and Track" model, which optionally adds noise and state tracking. The new estimated states are then processed through "Alerting and Guidance". An optional "Pilot Model" analyzes alerting and guidance output to determine if and when to maneuver the UAS to mitigate an alert. The "Navigation" model then executes any maneuvers created by the pilot model and forwards the updated flight to flight modeling. SaaControl employs various optimizations to improve system performance. These include gross filtering and longer time steps in trajectory prediction where feasible.

SaaControl simulation results are post processed to generate the desired DAA and safety metrics using a separate tool developed for the purpose.

Although SaaControl employs various optimizations and is quite flexible in the types of simulations possible and the ability to enable or disable some components, it has the following limitations for assessments of NAS-wide VFR data and UAS missions:

- Its behavior is fixed in code.
- It ingests and processes all input data even when encounters are unchanged between runs.
- It does not persist encounters in a standard format.
- It lacks a mechanism to control or select the types of encounters processed.

A typical one-day scenario used in SaaControl simulations includes 27,000 UAS flights and roughly 30,000 VFR flights. In contrast, for unmitigated simulations, the number of Losses of Well Clear for the entire day is on the order of 2,000 of which only 50 result in NMAC violations. This relative disparity demonstrates the need for more selective processing of NAS-wide data.

III. Encounter-Based Architecture

An alternative view of the JADEM architecture presented in Section II is based on a decomposition of the processing steps used in NAS-wide fast-time simulations. This decomposition is depicted in Fig. 2 and will henceforth be referred to as a "processing pipeline". The figure is simplified to depict unmitigated simulations, hence does not include pilot or navigation models, and omits miscellaneous other optimizations. Aside from the post processing step, there is a one to one correspondence between the pipeline stages in Fig. 2 and the functions in Fig. 1b. Despite the apparent similarity to Fig. 1b, the focus here shifts from the temporal processing indicated by the loop in Fig. 1b to encounter processing. However, this pipeline remains fixed in the SaaControl implementation. The new architecture described herein opens up this pipeline and makes it flexible by refactoring the processing steps into independent modules. This allows users to eliminate wasteful processing and enables omission or insertion of processing steps and even changing order of steps to suit research needs.

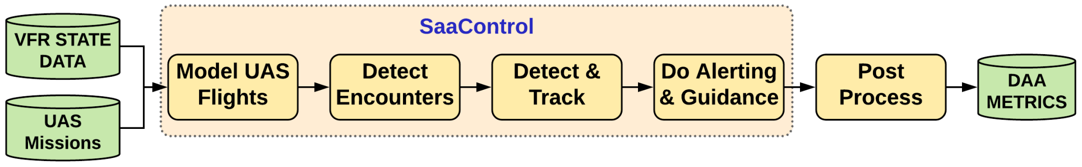


Fig. 2 SaaControl-based simulations presented as a processing pipeline

Furthermore, by modularizing the processing pipeline, we allow it to be composited from available modules or alternative module implementations. Figure 3 depicts one realization of a modular processing pipeline that applies a scenario selection module prior to running DAA with various settings for a parametric or trade-space study. More examples are provided in Section VII.C. As indicated in the figure, we persist and load data in between all processing steps, although this is not a requirement. UAS missions are modeled using a flight mechanics model and saved as trajectory states. VFR and UAS states are then processed to detect encounters and compute encounter properties. The resulting encounters are subsequently perturbed by the "Detect and Track" module. In the scenario selection step, a subset of the resulting perturbed encounters are selected for DAA processing (see Section VII for details on how this is accomplished). Finally, the DAA data are processed to generate metrics of interest. UAS mission modeling, encounter generation, perturbations, and scenario selection are executed once, whereas DAA can be executed multiple times for different configurations. Note that the "Compute Properties" and "Select Scenario" modules are newly developed capabilities. The "Model UAS Flights", "Detect Encounters", and "Detect and Track" modules are refactored from existing JADEM logic and enhanced in functionality.

This paper is focused on applications of the pipeline to unmitigated simulations, for which encounters are independent of one another; however, the new architecture can be applied to mitigated simulations with modest changes to the pipeline and/or its modules.

A. Definitions

This section provides definitions that will be used throughout the paper.

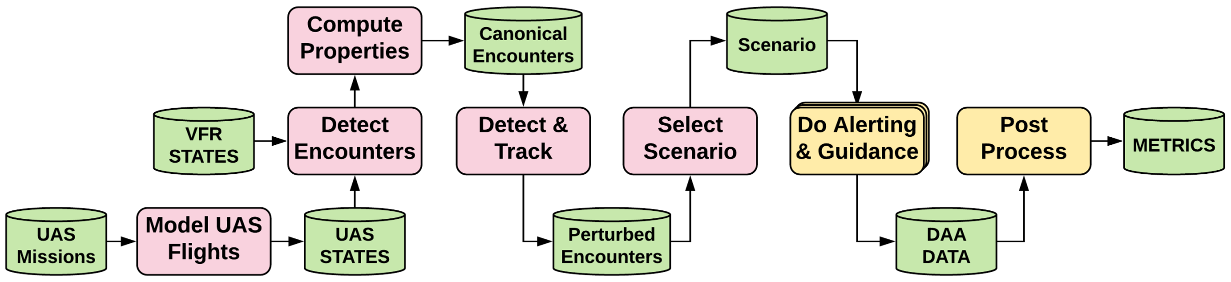


Fig. 3 Modular processing pipeline

1. Encounters

For our purposes, we define encounters to include trajectory portions from one ownship and one or more intruders that can potentially cause loss of well clear, an alert, or peripheral guidance [2] with the ownship. Peripheral guidance informs the pilot in control of a UAS of 'directions' to avoid in order to maintain separation from intruders that are not currently alerted [2, §2.2.4.4]. In addition to traffic data, a set of properties describing encounter characteristics of interest are also included in the encounter definition (see Section VI for details). Encounter properties are a unique new feature that facilitates selective processing of encounters as dictated by research needs (see Sections VII.A and VII.B). Note that an encounter, as defined here, includes one ownship only, so multiple ownship coordinated maneuvers are not modeled. A "canonical" encounter is one that is generated from source data such as VFR traffic and UAS data.

Note that an encounter's ownship trajectory data can be represented as a set of trajectory states or as a flight plan, which is typically a portion of the full flight plan. In most simulations, trajectory state data are sufficient. However, if a simulation requires mission recapture, then the relevant portion of the flight plan must also be included in the encounter.

2. Pairwise Encounters

The encounter definition given above is quite general but for many foundational studies, the interaction with multiple intruders is of less interest; instead single intruder interactions are desired. For these cases, encounters composed of one ownship and one intruder are more appropriate. Such encounters are referred to as pairwise encounters. Note that multi-intruder encounters can be created by composition of pairwise encounters of the same ownship that overlap temporally. Pairwise encounters can therefore be considered as fundamental building blocks for creating more general or complex encounters.

3. Scenarios

A scenario is a means to specify the encounters to be processed and evaluated by DAA simulations. It is defined as the set of encounters that satisfy specified criteria. For example, a scenario may include only those encounters whose ownship speed is within a specified range. Scenarios are typically generated by selecting from one or more sets of encounters such as those obtained from data for different days. Scenarios can also be created by filtering encounters from other scenarios. In addition to the list of encounters, a scenario specification includes a reference to the source data and the encounter selection criteria used to generate the scenario. The source data can be another scenario, multiple scenarios, or the set of VFR and UAS data. Referencing source data and selection criteria in the scenario specification provide traceability of scenario data; this is depicted in Fig. 4.

IV. UAS Flight Modeling

The operation of the flight modeling module is straightforward. It loads UAS mission data, models their flight trajectories, and persists the resulting trajectory data in several optional formats, which specify time series of positions and velocities. The module can use any flight physics model that satisfies JADEM's trajectory prediction interface. The default implementation uses JADEM's kinematic Multi-modal Adaptable Trajectory Generator, which can handle horizontal, vertical, and speed constraints and combinations thereof.

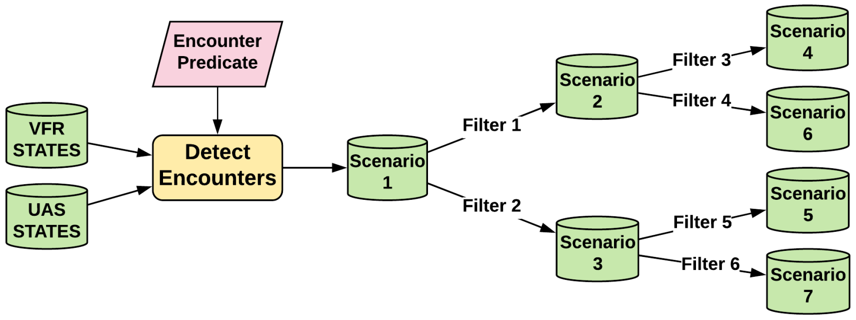


Fig. 4 Scenario Tree

UAS mission data are available with departures starting at 8AM GMT time and ending by 8AM 24 hours later. This means that for the first few hours of simulation UAS traffic is very sparse. It also means that flights departing late will be truncated at the 24-hour mark since typical simulations model 24 hours of VFR traffic. This is undesirable since the traffic density is distorted as a function of time during the day. The UAS Flight Modeling module provides an option to mitigate this by re-timing trajectory data that exceed 24 hours to the start of the day. It does so by subtracting 24 hours from their state times. This is equivalent to saying that the same UAS traffic missions are launched daily.

V. Encounter Detection

The goal of the core encounter detection algorithm is to extract encounters from input ownship and intruder flight data. The encounter-based architecture kept largely the same encounter detection algorithm that was used in JADEM for NAS-wide simulations [18]. The main change was converting this algorithm into an independent module.

A. Encounter Detection Criteria

Ownship and intruder flights for an encounter are selected by applying user-defined predicates (boolean expressions) to pairs of ownship and intruder states. The simplest predicate is a cylindrical "hockey puck" (a static disc). If this predicate is used, the encounter is determined by comparing horizontal distance d_{horiz} and vertical distance d_{vert} between ownship and intruder states for the same time with specified horizontal and vertical separation thresholds d^*_{horiz} and d^*_{vert}.

$$(d_{horiz} < d^*_{horiz}) \wedge (d_{vert} < d^*_{vert}) \tag{1}$$

d^*_{horiz}, d^*_{vert} in condition (1) should be large enough to ensure that any alerts and well clear violations will result in an encounter. Figure 5 shows one of the alerting structures that has been evaluated for UAS. The Buffered Well Clear Criteria (separation standards) in this table include a combination of Horizontal Miss Distance (HMD), vertical separation, and modified tau-separation. The time to Closest point of Approach (CPA) t_{CPA} from the Alerting Time Threshold column is more conservative and easier to use to estimate the minimal acceptable values for d^*_{horiz} and d^*_{vert}.

The largest distance between ownship and intruder causing an alert corresponds to the situation of a head-on encounter. This distance can be estimated as a product of approach speed ΔV_{horiz}, which is an intruder's speed relative to ownship, and t_{CPA}. In simulations considered in this paper, the air speed of UAS (ownship) and VFR traffic (intruders) does not exceed 300 knots. For a head-on encounter, this corresponds to $\Delta V_{horiz} < 600$ knots. The largest t_{CPA} in Alerting Time Threshold column for Preventive or Corrective Alert (Fig. 5) is 90 seconds. Therefore, the minimal value for $d^*_{horiz} = \Delta V_{horiz} \cdot t_{CPA} = 15$ nmi.

A similar estimate based on assumption of relative vertical speed ΔV_{vert} not exceeding 6000 fpm leads to the minimal value for $d^*_{vert} = \Delta V_{vert} \cdot t_{CPA} = 9000$ ft.

These are the minimal values for the alerting structure shown in Fig. 5. Typically, larger values are used to account for noisy data.

Symbol	Name	Buffered Well Clear Criteria	Alerting Time Threshold	Aural Alert
▲	Warning Alert	*DMOD* = *HMD** = 0.75 nmi Z_{THR} = 450 ft T_{mod} = 35 s	25 s (t_{CPA} ~ 60 s)	"Traffic, Maneuver Now"
●	Corrective Alert	*DMOD* = *HMD** = 0.75 nmi Z_{THR} = 450 ft T_{mod} = 35 s	55 s (t_{CPA} ~ 90 s)	"Traffic, Avoid"
○	Preventive Alert	*DMOD* = *HMD** = 1.0 nmi Z_{THR} = 700 ft T_{mod} = 35 s	55 s (t_{CPA} ~ 90 s)	"Traffic, Monitor"
△	Traffic	In surveillance field of regard	–	–

Fig. 5 An example of alerting structure

B. Encounter Detection Algorithm

The encounter detection algorithm uses ownship and intruder trajectories represented as a time series of position and velocity data. These trajectories may start and end at any time and need not be contiguous; therefore, data gaps are allowed.

The algorithm proceeds as follows (see Fig. 6):

1) Load trajectory data.
2) Prefilter aircraft states to reduce the amount of computations.
3) Apply predicates to identify encounters.
4) Post-filter to remove unusable encounters such as those with very short duration.
5) Persist remaining encounters.

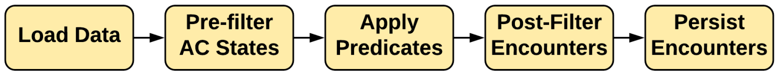


Fig. 6 Encounter Detection

Prefiltering is done in two steps. The goal of first step (the coarse filter) is to quickly identify the intruders, which could in principle come close to each ownship within a specified look-ahead time. This is done by mapping all initial intruder horizontal positions to a fixed horizontal grid, as described in [18].

Further prefiltering is done by skipping time-steps on the remaining flights. The amount of time to skip is estimated from the time it would take the ownship and intruder to be within the horizontal and vertical thresholds in the worst case if they have instantly turned to a collision course. Note that the amount of time that can be skipped, and hence the number of computations at prefiltering step, does not depend on the time step.

Time skipping ends when the computed skip interval is close to the time step. All sequential intruder states that satisfy the condition defined by Eq. (1) are included in the encounter for this intruder. Multi-intruder encounters may include more than one intruder if there is a time overlap between trajectory segments of different intruders that satisfy the condition of Eq. (1) for the same ownship.

The remaining encounters are persisted to the data store along with a corresponding scenario specification (see Section VII.B for more details).

VI. Encounter Properties

A number of encounter properties are calculated after encounter detection; the properties are persisted for each encounter. These properties can be used for encounter categorization and filtering as described in more detail in Section VII.A. This is an enabling feature of the new architecture that allows researchers to dramatically reduce the number of encounters that need to be processed. The most important and useful encounter properties are listed below.

EncounterID	a unique numeric identifier of this encounter
OwnshipAcID	the callsign of ownship flight
OwnshipAcType	the aircraft type of ownship used to determine its performance in trajectory generation process
EncounterDuration	encounter duration defined as the difference between `EndTime` and `StartTime`
Intruder.IntruderAcID	a unique string identifier for each intruder
Intruder.OwnshipSpeedAtCpa	ownship speed at CPA
Intruder.OwnshipAltitudeAtCpa	ownship altitude at CPA
Intruder.IntruderSpeedAtCpa	intruder speed at CPA
Intruder.IntruderAltitudeAtCpa	intruder altitude at CPA
Intruder.MinHmd	minimum Horizontal Miss Distance (HMD) [18] over encounter duration
Intruder.MinVmd	minimum Vertical Miss Distance (VMD) over encounter duration; the VMD is defined here as zero for vertically converging flights and as altitude difference between ownship and intruder states otherwise
Intruder.ViolatedNmac	indicates whether an intruder is within the NMAC volume defined as $(d_{horiz} < 500ft) \wedge (d_{vert} < 100ft)$

The "." notation is used to indicate grouping of related properties. In particular, all intruder-specific properties are grouped by intruder.

In addition to these "general properties" that do not depend on specific separation standards, several "DAA properties" can be defined, such as the properties of Loss of DAA Well Clear (LoDWC) and alerts of different "levels" corresponding to a particular alerting structure, such as shown in Fig. 5. These DAA properties are calculated by processing results generated by DAA. The processing pipeline for generating all encounter properties is shown in Fig. 7.

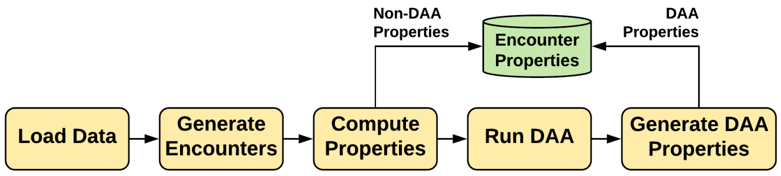


Fig. 7 Encounter Properties Generation

The next section describes how encounter properties are used to run efficient targeted simulations for NAS-wide assessments.

VII. Encounter Property Usage and Benefits: Filtering and Scenarios

The amount of data available for NAS-wide assessments is significantly larger than the number of events of interest (see Section II above). The ability to reduce NAS-wide data a priori is therefore essential for efficient processing. The remainder of this section demonstrates how this can be accomplished.

A. Encounter Filtering

The main use of encounter properties is to select the encounters of interest for a particular study or application. The technique used for this involves creating user-defined "filters" and is referenced herein as "encounter filtering".

A filter is defined as one or more predicates (boolean expressions). Each predicate compares a property specified via its path in the property files to a constant (threshold) or list of constants with appropriate units. The supported comparison operators are equality, strict and non-strict inequality, and list containment operators. List containment operators are used to check whether or not the property has one of several specified values defined in a list. The predicates can be combined using AND and OR logical operators.

For instance, a user may define a filter to select encounters for a particular `OwnshipAcType`. Another example would be selecting all encounters with `OwnshipAltitudeAtCpa` below a certain altitude in feet, `EncounterDuration` above a specified threshold in seconds, and `ViolatedNmac` = *false* (meaning the encounters do not result in NMAC). One especially useful filter selects encounters with `Intruder.MinHmd` and `Intruder.MinVmd` properties that do not exceed HMD^* and Z_{THR} thresholds (Fig. 5), since only these encounters can alert or result in LoDWC.

Applying the filter to a scenario results in another scenario that includes only the subset of encounters that passed the filter. This process is shown schematically in Fig. 8.

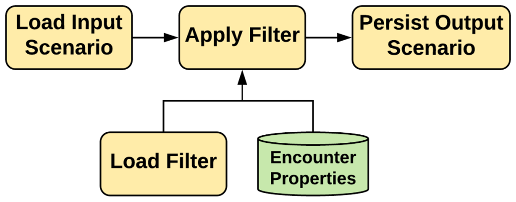


Fig. 8 Encounter Filtering

B. Scenario Selection

This section describes how the encounter detection and filtering methodology is typically used in simulations (see Fig. 4 for a depiction of this process).

The first step is to generate scenarios for available VFR and UAS data sets. These scenarios include all pairwise canonical encounters detected using a conservative predicate. As a result, these scenarios typically include many more encounters than are strictly required for a particular study. For instance, many encounters may not result in any events of interest, such as alert, LoDWC, or NMAC.

To "tailor" scenarios to a study, researchers create filters to be used to select encounters of interest. Filters may be defined using DAA independent properties, such as minimum HMD or speed and altitude at CPA or whether NMAC is violated. Filters may also be defined using properties that are specific to a DWC definition, such as maximum alert level or if an LoDWC exists. These properties are generated by running a scenario through DAA as shown in Fig. 7. In this case, DAA is typically run with conservative settings that form a superset of all the design parameters required for the study. The encounter filtering process shown in Fig. 8 can be repeated several times using different filters.

The resulting scenario is processed through DAA for each of the design parameters to generate alerting and guidance data. These data are subsequently processed to generate aggregate metrics for analysis and reporting. Note that the computationally expensive processes of canonical encounter generation and encounter property computation, need only be performed once; the resulting encounters can then be used for many different studies.

C. Use Cases and Benefits

The benefits of this approach can be illustrated by several use cases.

1) **Evaluating alternate DWC definitions**. In this case, the desired encounters are those that will generate an alert for at least one of the DWC definitions being evaluated. Therefore, the canonical encounters are processed once through DAA using a conservative DWC definition that covers all the desired DWC definitions. The resulting alerted encounters are then selected for use in the comparative study.

2) **Selecting Low SWaP encounters**. Low SWaP sensors are typically installed on relatively small and slow UAS flying below 10000 ft. Therefore, low SWaP encounters can be selected using filters composed from predicates on `OwnshipAcType`, `OwnshipSpeedAtCpa`, `OwnshipAltitudeAtCpa`, and possibly others, depending on specific goals of the study.
3) **Selecting flight-phase-specific encounters**. For studies focused on characterization of encounters by flight phase (e.g. level-level, level-climb, climb-descent, etc), encounters can be selected using `OwnshipFlightPhase` and `IntruderFlightPhase` properties. It should be noted, however, that determining `IntruderFlightPhase` is not trivial for noisy non-cooperative flight data.
4) **Validating canonical encounters**. To experimentally validate the encounters generated by the encounter detection process, one could compare results using the original less efficient approach to the results using canonical encounters generated for a test case. Should the results differ meaningfully, the encounter predicate is adjusted and the process repeated. This is depicted in Fig. 9.

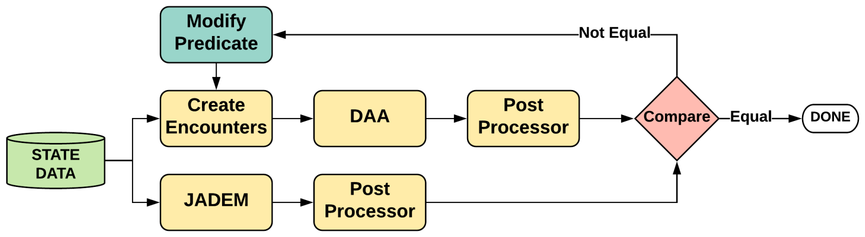


Fig. 9 Validating Canonical Encounters

VIII. Performance Comparison

This section compares the performance of the encounter-based approach to that of the original approach, which modeled flights from departure to destination. The simulations used in this comparison were done on multiprocessor Xeon Linux Servers with 128–512GB of memory.

In this section, Low SWaP DWC refers collectively to four DWC definitions being evaluated for lighter UAS operating below 10,000 ft with typical speeds below 100 knots. These definitions were selected from a larger pool of candidates investigated in [17]. Phase 1 DWC refers to the definition adopted for Phase 1 UAS MOPS and which, with its larger thresholds, is a superset of all the Low SWaP DWC definitions.

The comparisons were performed using 21 days of VFR traffic with a parametric suite of 96 individual configurations each. The baseline computation times, using the original approach, were estimated from an average day using one representative configuration rather than from the full suite of simulation runs. The processing pipeline used in this comparison study is depicted in Fig. 10.

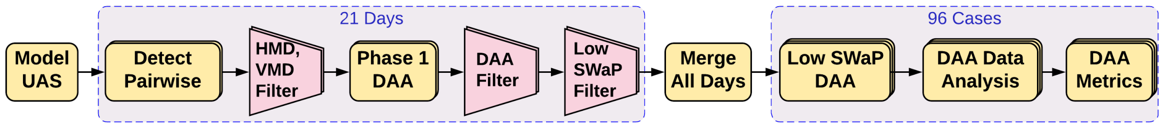


Fig. 10 Experiment Pipeline

The overall approach can be summarized in several steps:
1) Generate trajectories for all UAS flights.
2) Detect pairwise encounters.
3) Filter out all encounters that will not violate any of the Low SWaP DWC definitions; this is accomplished through the use of Phase 1 DWC, which covers all the Low SWaP DWC definitions.

4) Combine the resulting encounters into a single scenario.
5) Process the scenario through DAA
6) Post process DAA results and generate the metrics.

Trajectory generation uses the model described in Section IV. The flight times are subsequently adjusted for each simulation day. For each of the 21 days, pairwise encounters are detected using d^*_{horiz} = 20 nmi and d^*_{vert} = 10,000 ft, and their properties are computed. The main properties of interest to this study include HMD, VMD, and UAS and VFR speeds and altitudes at CPA.

Encounters that will have no chance of violating Low SWaP DWC are filtered out as follows:

1) *HMD & VMD Filter:* Create a filter that compares an encounter's minimum HMD and VMD against the HMD and Vertical Separation thresholds given in Phase 1 DWC. Use the filter to remove encounters whose minimum HMD and VMD exceed the corresponding thresholds and create a scenario of the remaining encounters.
2) *Phase 1 DAA:* Process the resulting scenario through DAA configured for Phase 1 DWC alerting. This step does not compute guidance since subsequent filtering requires only alert information.
3) *DAA Filter:* Use the alert results to create a scenario containing only those encounters that alert on Phase 1 DWC.
4) *Low SWaP Filter:* Apply a speed and altitude filter on the resulting scenario to remove all encounters whose speeds and altitudes exceed the Low SWaP performance limits. The resulting scenario will include only Low SWaP encounters that alert on Phase 1 DWC.

The resulting scenarios for the 21 days are small enough that they can be combined into a single scenario. This is done in order to reduce the number of simulations, since we are only interested in the overall results not those of individual days. As a result, we only have to contend with 96 instead of 2,016 (96 × 21) simulations.

Table 2 summarizes the salient computational performance figures. Performance of the new pipeline is presented in the top part of the table (all but the last three rows). For each step in the pipeline, the table reports the average time per run, the number of runs, the total time for all runs, and where applicable, the average time per encounter.

Table 2 Computational Performance Metrics

Processing Step	Average Time (hr)	# of runs	Total Time (hr)	Time (ms) Per Encounter
Model UAS	0.75	1	0.75	–
Detect Pairwise	1.23	21 days	25.9	9.6
HMD/VMD Filter	0.42	21 days	8.8	3.3
Phase 1 DAA	1.93	21 days	40.6	69
DAA Filter	0.14	21 days	2.9	5.0
Low SWaP Filter	0.0	21 days	0.0	0.7
Merge All Days	0.0	1 scenario	0.0	0.0
Low SWaP DAA	3.60	96 cases	346	157
DAA Data Analysis	8.69	96 cases	834	379
DAA Metrics	0.00	96 cases	0.2	0.1
Pipeline Total	–	–	**1,259**	–
Original DAA Processing	1.8	21 × 96	3,651	–
Original DAA Data Analysis	1.5	21 × 96	3,024	–
Original Approach Total	–	–	**6,675**	–

Inspection of the times consumed by each step in the pipeline shows that "Low SWaP DAA" and "DAA Data Analysis" have the most impact on overall processing time. Therefore reducing the number of encounters that need to be processed by these two steps is critical. The processing steps preceding "Low SWaP DAA" prove to be an effective way to do so, with relatively modest impact on overall performance.

One thing of note is that the average time per encounter for the "Low SWaP DAA" pipeline step is significantly larger than that for the "Phase 1 DAA" pipeline step. This is mainly due to guidance computations, which are applied in "Low SWaP DAA" but not in "Phase 1 DAA".

Also reported in Table 2 are the estimated times (the last three rows) had the simulations been run using JADEM as shown in Fig. 2. These time estimates were obtained from a single run and scaled to all 2,016 runs.

The main figure of merit in the table is the total processing time, which is shown to be significantly reduced using the new architecture.

Table 3 summarizes the impact of filtering on the volume of data to be processed. Filtering reduces the amount of data to be processed a hundredfold from nearly 10,000,000 to just over 80,000. The major impact is seen to result from filtering for Phase 1 DWC-alerted encounters using the "DAA Filter".

Table 3 Filtering Performance

Processing Step	Total Input Encounters	Total Output Encounters
Detect Pairwise	–	9,692,090
HMD/VMD Filter	9,692,090	2,106,610
DAA Filter	2,106,610	129,394
Low SWaP Filter	129,394	82,524

The main takeaways from the two tables are that filtering is key to improving the overall performance and that using the new processing pipeline reduced the total computation time five-fold.

Note that all I/O operations were to the file system, with I/O over the network consuming approximately twice as much time as local I/O. In the early stages of the pipeline, where the data volumes are large, the processing time is dominated by I/O. This is especially true for encounter detection where computations consumed only 20% of the total time. Note also that all calculations were performed without any parallelization, but they can be parallelized in a number of ways. The new architecture makes parallelization especially easy, since each encounter can be processed independently.

IX. Summary and Concluding Remarks

JADEM is an effective tool for conducting fast-time parametric and trade-space studies in support of UAS integration in the NAS. NASA studies use NAS-wide VFR data and projected UAS missions. Prior NAS-wide simulations were conducted on the full airspace data, modeling flights from departure to destination. As a result, simulations were time consuming, required substantial computational resources, and were wasteful since the events of interest represent a small fraction of flight time.

Furthermore, research studies often required selective processing of encounters but the existing tools did not provide mechanisms for selecting encounters a priori. Finally, sharing the voluminous NAS-wide data with our partners was cumbersome.

To address these issues, JADEM was refactored to an encounter-centric processing pipeline composed of a set of decoupled modules. This has allowed simulations to be conducted on an encounter-centric basis. This approach has been shown to have several advantages, which include reduced memory footprint and processing time, flexible scenario generation, a flexible DAA processing pipeline, and standardized encounter and scenario representations to name a few. Reduced computational time enabled trade-space studies to be conducted with finer granularity and better trade-space coverage.

The architecture described in this paper can be enhanced in several ways. New features will be added to support upcoming research requirements such as evaluating the impact of alert mitigation. This would require the inclusion of a feedback loop in the processing pipeline to support avoidance maneuvers. In addition, further computational performance enhancements are possible including the use of scalable and big data architectures.

Acknowledgments

We wish to express gratitude to Confesor Santiago for his concept contributions and our fellow DAA team developers for their tireless efforts to realize the encounter-based simulation architecture.

References

[1] *Integration of Civil Unmanned Aircraft Systems (UAS) in the National Airspace System (NAS) Roadmap*, 2^{nd} ed., Federal Aviation Administration, Washington, DC, 2018.

[2] SC-228, *Detect and Avoid Minimum Operational Performance Standards Phase I (DAA MOPS)*, RTCA, Washington, DC, 2017.

[3] Weinert, A. J., Harkleroad, E. P., Grith, J., Edwards, M. W., and Kochenderfer, M. J., "Uncorrelated Encounter Model of the National Airspace System, Version 2.0," Tech. Rep. ATC-404, MIT Lincoln Laboratory, Lexington, Massachusetts, August 2013.

[4] Weibel, R. E., Edwards, M. W. M., and Fernandes, C. S., "Establishing a Risk-Based Separation Standard for Unmanned Aircraft Self Separation," *Ninth USA/Europe Air Traffic Management Research & Development Seminar, ATM2011*, ATM Seminar, Berlin, Germany, 2011.

[5] Edwards, M. W. M., and Mackay, J. K., "Determining Required Surveillance Performance for Unmanned Aircraft Sense and Avoid," *17th AIAA Aviation Technology, Integration, and Operations Conference (AIAA 2017-4385)*, AIAA Aviation Forum, Denver, Colorado, 2017.

[6] Lee, R., Kochenderfer, M. J., Mengshoel, O. J., Brat, G. P., and Owen, M. P., "Adaptive Stress Testing of Airborne Collision Avoidance Systems," *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*, IEEE, Prague, Czech, 2015.

[7] Hardy, J., Jack, D. P., and Hoffler, K. D., "Sensitivity Analysis of Detect and Avoid Well Clear Parameter Variations on UAS DAA Sensor Requirements," *2018 Aviation Technology, Integration, and Operations Conference*, AIAA Aviation Forum, Atlanta, Georgia, 2018.

[8] Lee, S., Park, C., Thipphavong, D., Isaacson, D., and Santiago, C., "Evaluating Alerting and Resolution Performance of a UAS Detect-And-Avoid (DAA) System," Tech. Rep. NASA/TM-2016-219067, NASA Ames Research Center, Moffett Field, California, February 2016.

[9] Ayyalasomayajula, S., Wieland, F., Trani, A., and Hinze, N., "Unmanned Aircraft System Demand Generation and Airspace Performance Impact Prediction," *Proceedings of the 32nd IEEE Digital Avionics Systems Conference*, IEEE, Syracuse, NY, 2013.

[10] Park, C., Lee, H., and Musaffar, B., "Radar Data Tracking Using Minimum Spanning Tree-Based Clustering Algorithm," *11th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference (AIAA-2011-6825)*, AIAA, Virginia Beach, VA, 2011.

[11] Lee, S., Park, C., Johnson, M., and Mueller, E., "Investigating Effects of "Well Clear" Definitions on UAS Sense-And-Avoid Operations," *2013 AIAA Aviation Technology, Integration, and Operations Conference*, AIAA, Los Angeles, CA, 2013.

[12] Park, C., Lee, S., and Mueller, E., "Investigating Detect-and-Avoid Surveillance Performance for Unmanned Aircraft Systems," *2014 AIAA Aviation Technology, Integration, and Operations Conference*, AIAA, Atlanta, GA, 2014.

[13] Johnson, M., Mueller, E., and Santiago, C., "Characteristics of a Well Clear Definition and Alerting Criteria for Encounters between UAS and Manned Aircraft in Class E Airspace," *10th USA/Europe ATM Research and Development Seminar, ATM2015*, ATM Seminar, Lisbon, Portugal, 2015.

[14] Cone, A., Thipphavong, D., Lee, S., and Santiago, C., "UAS Well Clear Recovery against Non-Cooperative Intruders using Vertical Maneuvers," *17th AIAA Aviation Technnology (AIAA-2017-4382)*, AIAA, Denver, Colorado, 2017.

[15] Thipphavong, D., Cone, A., and Lee, S., "Ensuring Interoperability between UAS Detect-and- Avoid and Manned Aircraft Collision Avoidance," *Twelfth USA/Europe Air Traffic Management Research and Development Seminar, ATM2017*, ATM Seminar, Seattle, Washington, 2017.

[16] Cook, S. P., Brooks, D., Cole, R., Hackenberg, D., and Raska, V., "Defining Well Clear for Unmanned Aircraft Systems," *AIAA Infotech @ Aerospace (AIAA 2015-0481)*, AIAA SciTech Forum, Kissimmee, Florida, 2015.

[17] Wu, M. G., Cone, A. C., Lee, S., Chen, C., Edwards, M. W., and Jack, D. P., "Well Clear Trade Study for Unmanned Aircraft System Detect And Avoid with Non-Cooperative Aircraft," *2018 Aviation Technology, Integration, and Operations Conference (AIAA 2018-2876)*, AIAA Aviation Forum, Atlanta, Georgia, 2018.

[18] Abramson, M., Refai, M., and Santiago, C., "The Generic Resolution Advisor and Conflict Evaluator (GRACE) for Unmanned Aircraft Detect-And-Avoid Systems," Tech. Rep. NASA/TM-2017-219507, NASA Ames Research Center, Moffett Field, California, 2017.