An Introduction to Fast Time Simulations for RPAS Collision Avoidance System Evaluation

Guido Manfredi, Yannick Jestin

An Introduction to Fast Time Simulations for RPAS Collision Avoidance System Evaluation

Guido Manfredi, Yannick Jestin
ENGIE Ineo - Groupe ADP - Safran RPAS Chair
Universite de Toulouse, Ecole Nationale d'Aviation Civile
Toulouse, France
Email: guido.manfredi@enac.fr

***Abstract*—Collision voidance systems are crucial for RPAS integration, yet comparing their performances remain difficult. We believe that using fast time simulations and standard evaluation metrics would facilitate their comparison while providing insight into their benefits. However, fast time simulations are often viewed as hard to set up and limited to large scale demonstrations. We believe even small experiments can take advantage of them with huge benefits. The aim of this work is to ease access to fast time simulations by providing explanations, examples and references to previous works and to free software. We also list commonly used evaluation metrics for collision avoidance system performance ranking. By easing the setup of fast time simulation experiments, we believe future works will be able to provide their results in a more detailed and comparable form.**

***Index Terms*—Introduction; Tutorial; Detect And Avoid; Collision Avoidance; Fast Time Simulation**

I. Introduction

According to the SESAR-JU European Drones Outlook Study [1], the European RPAS market is expected to weight 10 billion euros annually by 2035. The creation of such a large business requires an ongoing regulatory effort. However, before regulation can provide the rules, the research communities need to overcome some technological locks including the Detect And Avoid (DAA) problem. In the context of this work, DAA should be understood as aerial traffic DAA, which involves two functions: Traffic Avoidance (TrA), also called Remain Well Clear (RWC), and Collision Avoidance (CA). However, in controlled airspace, TrA is provided by ATC services so the rest of this paper focuses on the CA function only.

As will be shown in the next section, numerous works have proposed CA algorithms, however their comparison remains difficult because they use different evaluation methods. In this paper, we advocate in favor of using Fast Time Simulations (FTSs) with appropriate validation metrics to assess the effectiveness of a Collision Avoidance System (CAS). In order to ease the use of such method, section III provides an overview of the methodology and the main components of a FTS for CAS. These components need to be integrated in a simulation framework to analyze their interactions, so Section IV introduces three freely available simulation frameworks. Section V presents metrics to post process the FTS output into a performance assessment. Finally, Section VI concludes on the mains reasons to rely on FTS when evaluating a CAS and the main steps to do so.

II. Existing Evaluation Methods

Simulations and evaluations used in previous works provide useful insight into CAS evaluation methods and what needs to be simulated (flying objects, flight patterns, sensors, etc.). We considered three different sources of information: evaluation from the research communities to show the performances of a given algorithm, evaluation with large scale demonstrator for RPAS integration in civil airspace and evaluations by regulatory bodies to authorize or enforce the use of a CAS for manned aviation. Evaluation methods can be grouped in different categories as proposed by Lacher et al. in [2], where each family of methods addresses specific questions (see Table I). In the following, the term Fast Time Simulation (FTS) is used for simulations where all the agents are computer models implemented by programs with no real-time constraints. In the case of Real Time Simulations (RTS), part of the agents are computer models while some of them are human operators (e.g. pilots, controllers). Finally, in the case of test flights, one or more actual aircraft, equipped with the CAS to be evaluated, are flown along real or simulated traffic.

A. Algorithms Evaluation

Most research algorithm evaluations aim at showing the performances of the collision avoidance or the surveillance part on its own, they rarely consider the whole environment and all its uncertainties. As sensors are scarcely simulated, the first four lines of Table I are rarely considered in these kinds of evaluations. In numerous works, evaluation encounters are designed by hand, usually depicting challenging or worst case scenarios. In [3], the authors evaluate a CA method based on spiraling motion for avoidance. The evaluation is done on four manually defined scenarios representing challenging geometries, with a point mass model for aircraft. The authors of [4] perform a simulation with five-degree-of-freedom aircraft models for both ownship and intruders. Encounters are designed by hand and aim at representing typical collision encounters. In [5], the evaluation scenarios are handmade 2-D grids containing moving and/or fixed obstacles, all aircraft being considered as point masses.

TABLE I: Evaluation Methods, and the Questions That Can Be Addressed (Table from [2]). In this Table, the Fast Time Simulation methods are called Monte Carlo Simulations.

	Bench Test (component-level)	Bench Test (integrated System)	Field Data Collection (component-level)	Flight Test (integrated system)	Fault-tree / Hazard Analysis	Monte Carlo Simulations	Human-in-the Loop Laboratory Sims	Operational Evaluation	Performance Monitoring
What does the sensor see? Range? Azimuth? Elevation? Speed?	✓		✓						
How large an object can it see? How far away?	✓		✓						
Are sensor specifications accurate?	✓		✓						
Does the system function together?		✓		✓					
Does collision avoidance capability provide safe separation from manned aircraft?		✓		✓	✓		✓		
Does collision avoidance algorithms react in an acceptable way for other pilots?				✓			✓	✓	✓
Does capability act in an acceptable way in the context of the operating environment (e.g., ATM)?				✓			✓	✓	✓
What are the limits of the capability? Conditions? Size or number of targets?	✓	✓	✓	✓		✓			
How does collision avoidance technology compare to "see and avoid"?				✓	✓	✓	✓	✓	✓
What is the overall system performance? i.e., resulting collision risk					✓	✓	✓	✓	✓

In order to better assess the effectiveness of a CAS, some works perform FTS: the CAS is evaluated in thousands of simulated encounters. In [6], the CAS is evaluated by simulating a RPAS flying a pattern in the middle of surrounding traffic obtained through logged ADS-B data. In [7], the authors generate ten million encounters using the MIT uncorrelated encounter model [8]. Finally, Allignol et al. simulate more than two hundred thousand encounters based on records of real traffic data and simulated RPAS trajectories [9].

In some cases, RTS with a simulated or real flight have been used. The method proposed in [6] is evaluated with flight tests where the encounter geometries were manually defined. In [10], Lyu et al. use the FlightGear JSBsim simulator to integrate a simulated RPAS into real-time acquired ADS-B data.

When it comes to simple CA algorithms evaluation, most works rely on simulations designed by hand, rarely on FTS, RTS or test flights. These methods are more common for larger scale experiments as will be shown in the following.

B. Large Scale Evaluations for RPAS integration

Large scale projects, aimed at evaluating integrated solutions, rely on a large panel of validation tools. Though the following methods are not all completely dedicated to CAS, these evaluation methods are worth mentioning.

The ODREA project [11] included FTS, RTS and flight tests. The FTS relied on recorded traffic of one day over the south-west of France. These where replayed and RPAS were simulated as performing missions in the airspace with six different motion patterns, different offsets to the RPAS position and different equipage (TCAS/DAA, coordination) for a total of 874,800 encounters. The RTS involved actual Air Traffic Control Operators (ATCOs) and pseudo-pilots simulating the RPAS and background traffic. Three RPAS (2 slow, 1 fast) where simulated in the middle of traffic in nominal and degraded cases (loss of C2 link). Finally, flight tests involved take-off and landing from/to airports following different routes.

The MIDCAS project [12] aimed at gaining insight into the efficiency of a DAA solution for TrA and CA. The preliminary simulations consisted in creating a flight area where intruders were spawned at random locations, with fixed speed in order to reach a given density of aircraft. Simulated flights lasted 30s and where run to accumulate 100.000 hours. For the FTS, a Monte Carlo analysis has been carried out on 30.000 encounters generated from the European Encounter Model (primary radar data) and Uncorrelated Encounter Model. The RPAS is simulated as a 6 DOF object. It is worth noting that a pilot behavior model has been used in the simulations. Results were analyzed for statistics over NMAC and Target Level of Safety (TLS). For the RTS, specific short scenarios have been manually designed, with multiple intruders. The ATC simulations where performed by recreating flight plans across three active Airports and CTR in Sweden with flights durations over 40 minutes. The RPAS was commanded by a pseudo pilot and traffic and collision avoidance maneuvers where coordinated by actual ATC operators. Finally, actual flights tests were carried out with more than 90 scenarios (29 virtual and 71 real intruders) to test the traffic and collision avoidance. This work is currently extended in the MIDCAS Standardization Support Phase (MIDCAS SSP) with more FTS and RTS to come.

In a long series of experiments aimed at evaluating different CAS methods, the NASA performed numerous FTS, RTS, pilot and controller evaluations as well as test flights. Preliminary works allowed defining needed notions like sensors requirements [13] and a well-clear definition [14]. Based on these studies, RPAS integration experiments [15], tested three different DAA algorithms, CPDS (General Atomics and TU Delft), JADEM (NASA Ames) and Stratway+ (NASA Langley) during flight tests. These flight tests mixed control tower and background traffic simulated with the NASA LVC environment and two live aircraft for the ownship and the intruder. Through 11 flights, 108 encounters were created.

These examples show that various simulation types are usually combined for large scale evaluations, a common combination being: FTS, RTS and flight tests. As noted in the MIDCAS documentation, FTSs allow statistical validation, RTSs provide functionalities validation and flight tests give results for actual situations.

C. Evaluation for Regulation

CASs for manned aviation have been around for decades. Before being authorized or enforced, these systems have been thoroughly tested. In the last fifteen years, three important systems have been evaluated: ACAS II, STCA, and ACAS X (in the US only).

During the ACASA project [16], multiple experiments have been conducted to evaluate the effect of ACAS II on safety depending on various conditions. These evaluations have led to the development and use of the European Encounter model,

and focused on the number of unresolved/induced encounters as well as the risk ratio.

To establish Short Term Conflict Alert (STCA) performance requirements, a series of FTS have been carried out in [17]. Encounters were generated using the ATM encounter model (440,000 encounters where separation minima are preserved) and the safety encounter model (200,000 encounters with risk of Mid Air Collision).

The recent ACAS X preliminary evaluations were done through FTS followed by test flights. Radar surveillance from 100,000 real encounters that resulted in TCAS RA were used to compare to TCAS performances. To obtain more data, hundreds of thousands of encounters were generated using the correlated and uncorrelated MIT encounter models. Plus, specific models for specific situations (500ft vertical separation, 1000ft vertical separation, closely spaced parallel runways) were developed. Finally, encounters from real flights recorded before the use of TCAS served for stress cases. After tuning the algorithm with the preliminary tests, successive flight campaigns have been carried out to evaluate the system in real flight.

On top of these CAS evaluations, concepts of "well-clear" have been recently evaluated [18] and though this is not specifically CAS evaluation, the methodology is similar and so interesting to us. The idea is to define with objective metrics the subjective notion of "well-clear". Three candidate definition from NASA, MIT and AFRL were evaluated and their results compared. The evaluation was performed in three stages. First a Monte-Carlo simulation, then a human-in-the-loop simulation and finally a stress case analysis.

Works evaluating new CA algorithms most often rely on hand made worst cases scenarios to show the robustness of their methods. For RPAS large scale evaluations, different simulation methods, from handmade simulations to flight tests, are used to answers different questions about the CAS (see Table I). Finally, for regulation validation, it appears like FTS is the preferred evaluation method as it allows computing statistically significant results, the main drawback being that it requires an encounter model representative of the airspace being assessed. As stated by DeGarmo (MITRE) [19] : "How exactly UAV [RPA] operation will affect airspace capacity and traffic flows can only be answered through simulations of UAVs [RPAs] within specified airspace. Part of this simulation must account for how the vehicles will operate (speeds, altitudes, endurance), where they will operate (mission), and in what numbers".

The next section presents the main components of a CAS and the different environmental agents in the FTS that feed them information.

III. FTS FOR CAS

Before going on, we present a general architecture for a CAS and detail its sub-function. Then, we present the modules needed in the FTS to feed information to these sub-functions and simulate all the environment.

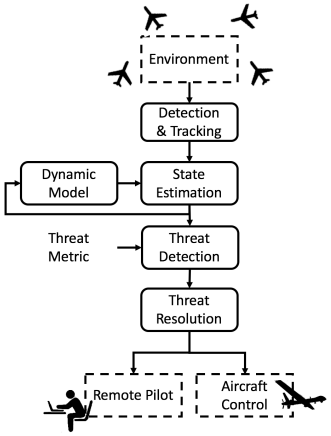

Fig. 1: The five main steps of a collision avoidance system.

A. Collision Avoidance System

In the following the aircraft equipped with the CAS is called the ownship, surrounding aircraft currently monitored are intruders and aircraft requiring avoidance are threats. The goal of a CAS is to prevent Near Mid-Air Collisions (NMACs), i.e. threats entering the ownship collision volume. In order to have time for the avoidance process, a collision avoidance threshold is defined around the collision volume.

To perform this avoidance, a CAS is composed of four different processes which allow processing surveillance data, estimating the current state and next state, detecting possible threats and finding a maneuver to avoid them. These steps are illustrated in Figure 1 and detailed in the following.

1) Detection & Tracking: The CAS surveillance system monitors aircraft within its sensors range. The detection and tracking functions collect sensor measurements to creates tracks. It also has the responsibility to merge the input from the sensors and verify the data provided by sensors deemed unsecure. Depending on the process and sensors involved, a confidence level is associated with each track.

2) State Estimation: From different data provided by the surveillance, the state estimation process computes a set of variables which describes the current situation w.r.t an intruder or a group of intruders. The number of variables should be kept low to reduce computational complexity but descriptive enough to provide a proper view of the current state. For example, in TCAS the state is describe by τ, an estimation of the time before Closest Point of Approach (CPA), and h, the relative altitude between ownship and a threat. A common method to reduce the number of state variables is to use aggregate variables which mix various information into a single variable, e.g. in TCAS τ is an aggregation of the range and closure rate variables. To handle noise in the surveillance data, modern methods consider the state variables as probability distributions.

3) Dynamic Model: The dynamic model allows estimating the future state of the world, at the next time step. It can include ownship/intruder dynamics and pilot response models. Estimation of an intruder's dynamic model can be computed with straight projections, worst case projections, probabilis-

tic projections or with a flight plan sharing approach. The dynamic model computation often depends on the previously estimated state to ensure a memory effect, where past states influence the estimation of future states.

4) Threat Metric: The threat metric is the formal definitions of the CA threshold. It can depend on the ownship-intruder distance, closure rate, time to CPA and/or time to collision volume violation. It can also consider only horizontal metrics or both horizontal and vertical metrics. Often, like in TCAS, the threat metric is a combination of both time and space measures with multiple conditions to be met before declaring an intruder has become a threat.

5) Threat Detection: For each monitored aircraft, the threat metric allows the threat detection process to estimates if there is a collision risk. Based on the current and projected states, and on a set of rules defining the collision avoidance threshold, it determines if an intruder has become a threat.

6) Threat Resolution: When one, or various, intruders are labelled as threats, the threat resolution module computes a maneuver to avoid them. This maneuver can influence three parameters: ownship speed, horizontal direction and vertical direction. The resulting maneuver can be a simple command to follow at a given time or a full trajectory leading out of danger.

The CAS output is communicated to the remote pilot and is executed by the RPA or by the remote pilot (depending on the remote pilot being in-the-loop or on-the-loop). After this quick overview of CAS architectures and workings, let's focus on the evaluation of such algorithms.

B. Fast Time Simulations

Fast Time Simulations (FTS) allow performing a high number of tests with limited logistics, in limited time and with no safety concerns. The different elements of the world need to be modeled, mainly the aircraft traffic and RPASs as well as the sensors used by each one of them. Encounters that will activate the CASs need to be generated. As will be discussed later, there are various means to do that with different objectives in mind.

1) Aircraft Models: There are three popular representations for aircraft. The simplest one is the point mass model, where an aircraft is considered as a point upon which forces are applied. This representation completely hides the flight mechanics. A more complex representation involves considering the aircraft as a 4 degree of freedom object [18]. Then an aerodynamic model of the aircraft is needed to represent its behavior depending on these DOFs. Finally, the most complex model uses 6 DOFs, the additional dimensions increase the realism and complexity of the simulation. For the two previous representations, manned aircraft models are available in the Base of Aircraft Data (BADA). Some RPAS models have been created and integrated as well in the BADA database [20]. The precision of the aircraft models need to be different depending on the considered constraints. For example, if wind needs to be accounted for, then the point mass model provides a rough approximation. It is important to note that all the aircraft involved in the evaluation do not need precise modelling. A good compromise can be reached by representing the traffic with the point mass model and to have the RPAS modeled with 4 to 6 DOFs.

The model should include as a minimum the following information: maximum climb rate, vertical acceleration, maximum turn rate, and maximum turn acceleration.

2) Pilot Models: The reaction time and decisions of the remote or intruder aircraft pilot can be modeled to increase the simulation's realism. It provides the advisory response rate, and advisory response delay. The simplest model is the procedural pilot, it considers that the pilot follows all the required procedures for his flight without accounting for the surrounding aircraft, except if its onboard system (e.g. TCAS) gives an alert. In this model, no reaction to a maneuvering aircraft will be done. Variations of this model consider the reaction time of a pilot. It can be instantaneous, though it has been shown that this creates discrepancies with real data [21]. It can take a fixed amount of time, e.g. ACAS considers that pilots answer resolutions in 5s [22]. Or it can use a probabilistic reaction time model, as is done in ACAS X [23]. Advanced pilot models have been developed, where the pilot's reaction changes depending on the maneuver of surrounding aircraft. This has been modeled using game theoretic models in [24].

3) Sensor Models: Like for the aircraft models, the sensors can be represented with different levels of detail. The simplest level is to consider the suite of sensors as a whole. For example, in [13] the suite of sensors is represented by a single sensor with given range, Field of Regard (FOR) and update rate.

A second approach is to model each sensor independently. This requires the existence of a surveillance module able to perform data fusion to combine the information brought by each sensor. Care need to be taken for dependent sensors to model their source of information. Indeed, the GNSS data used by a plane to estimate its own position need to have the same error as the one it is broadcasting through its ADS-B. Modeling sensors independently has the benefit of differentiating cooperative sensors, i.e. sensors taking advantage of intruder aircraft avionics, from non-cooperative sensors, i.e. sensors relying only on onboard processing for aircraft detection. In the following, we provide for common RPAS sensors, simple models for the GNSS and INS as well as models developed in [10] for the Air-To-Air-Radar (ATAR) and Electro Optics/Infra Red (EO/IR).

All aircraft are equipped with Global Navigation Satellite System (GNSS) receivers to acquire their localization, latitude, longitude, altitude, in a global frame. Modeling of a GNSS system can reach high levels of complexity. However, for aircraft positioning such precision is neither required nor desirable as it would be far too complex to simulate. Instead, a simple model consists in considering the three dimensions of the localization as independent and adding Gaussian noise to each of them. Though, this approach does not consider the time correlation between successive measurements, so a more realistic representation can be achieved using a first order Gauss-Markov process, as described in [25]. According to this

work,

$$y(n) = ay(n-1) + u(n),$$

where n is the time (discrete), $y(n)$ is the GNSS measurement at time n, a is a coefficient belonging to $]0;1[$ and $u(n)$ is a random variable that follows a Gaussian distribution. While remaining simple, this method provides a more realistic GNSS representation.

In order to estimate their attitude, aircraft rely on Inertial Navigation System (INS) measurements combined with estimation filters to track their motion. This provides roll, pitch, yaw, the corresponding angular speeds as well as ground speed and vertical speed. Integrating this data allows computing the aircraft localization. Again, the simulation of an INS is the topic of a fair amount of research. Some of the most advanced models take into account the equation of motion of the INS parts to model it, but again this is too complex for the task at hand. In the following, we propose a model combining white noise, a constant bias and first order Markov noise. The resulting noise is computed as follows,

$$\eta = \sigma(1 + 0.5randn(1 + \sqrt{\frac{1-\beta}{1+\beta}})/\sqrt{3})$$

with σ the quadratic error at 95%, $randn$ a random number from a Gaussian distribution and β a Markov noise constant defined as

$$\beta = \exp\frac{-\delta_t}{\tau}$$

where δ is the sampling period and τ a constant specific to the INS. The previous noise is applied independently on the eight parameters provided by the INS.

The Air-To-Air-Radar (ATAR) is a main sensor for medium and large RPAS as it is the only non-cooperative sensor working in IMC while having large FOR and range. According to [10], an ATAR model can be simplified by neglecting the signal processing part. Their model consists in projecting the intruder global position X_i into a spherical coordinates system S centered on the ownship global position X_o and adding zero mean Gaussian noise independently on the three dimensions. With $X_{io} = X_i - X_o = (x_{io}, y_{io}, z_{io})$ and $S_{io} = (r_{io}, \alpha_{io}, \beta_{io})$ the expression of X_{io} in S:

$$\begin{aligned}
r_{io} &= \sqrt{x_{io}^2 + y_{io}^2 + z_{io}^2} + \eta_r, & \eta_r &\sim N(0, \sigma_r) \\
\theta_{io} &= \arctan\frac{y_{io}}{x_{io}} + \eta_\theta, & \eta_\theta &\sim N(0, \sigma_\theta) \\
\phi_{io} &= \arccos\frac{z_{io}}{r_{io}} + \eta_\phi, & \eta_\phi &\sim N(0, \sigma_\phi)
\end{aligned}$$

where $(\eta_r, \eta_\theta, \eta_\phi)$ represents the Gaussian noise. So the output of the ATAR is the $(r_{io}, \alpha_{io}, \beta_{io})$ coordinates with associated rates and variances.

Because it is a cheap sensor that allows a great deal of different processes, Electro Optics and Infra Red (EO/IR) are generally present in the RPAS sensor suite. In order to operate by day and by night, visual spectrum cameras are coupled with Infra Red cameras. Simulating such sensors in detail would require a photorealistic simulator able to generate realistic synthetic images of the simulation environment. This can be done thanks to COTS software, like in [10] where the authors used the Vega Prime software [26] to simulate both EO and IR sensors. Proper simulation of the IR sensor implies some more processing than EO and a possible method is described in [27]. Once images are simulated, image processing techniques, like the ones described in [28], can be used to retrieve the intruder bearing and relative altitude. However, retrieving precise range would require a complicated setup which is rarely considered.

Simulated sensors introduces noise in the traffic position measurements which need to be handled with a probabilistic representation of the environment in the CASs to be evaluated. In order to keep the noise in realistic boundaries, it is crucial to consider the minimum requirement on some equipment. For example, simulated ADS-B data should conforme to the minimum navigation precision described in existing standards (NIC, NACp, NACv, SIL levels). Similarly, active surveillance (ACAS) should conforme to ACAS's MOPS (DO-185B). With models for aircraft, RPAS and their sensors, it is time to proceed to the encounter models, i.e. how to simulate conflicts between a RPAS and traffic in a realistic fashion.

4) Encounter Model: The encounter model defines the parameters of the conflict between a RPAS and one or various aircraft. The objective is to create realistic encounters while covering as many cases as possible. Depending on the goal of the evaluation, different types of models can be used: safety critical (for safety analysis), operational suitability (for acceptability), stressing (for late stage evaluations), specific (to be sure the system can handle some precise scenarios). Multiple encounter models have been built for manned aviation by analyzing radar tracks (see Table II). However, this is not possible in the RPAS case due to the lack of data concerning how RPAS fly in the airspace. Nevertheless, we will present three different approaches to deal with the encounter modeling problem.

First, the parameterized methods rely on hand designed encounters with a parameterization done by experts. This type of approach is useful to define stress cases which are unlikely to happen during an actual flight but need to be handled nonetheless. The AFRL has defined such stress case encounters when evaluating "well clear" definitions [18]. Because of the limited number of encounters that can be designed manually, this approach offers the possibility to examine in fine detail each encounter and its development. This is especially useful to get an insight of the CAS behavior in abnormal and emergency situations. As an example, the authors of [32] propose to generate encounters with evolutionary computation methods to create difficult cases.

The probabilistic approach consists in using real traffic data to estimate a probability distribution over the encounter parameters. Once this distribution is estimated, generating an encounter is as simple as sampling the distribution. This approach is often called "Monte Carlo" simulation, though we prefer to avoid this name as the Monte Carlo method can be used to compute different type of data in simulations.

TABLE II: Existing encounter models for manned civil aviation, built from primary radar tracks.

Encounter Model	Number of Encounters	Encounter Type	Year
ACASA [16]	1243 FR + 1144 UK	Encounters likely to trigger TCAS RA	2002
ASAS [29]	147,250 EU	Separation minima are generally preserved, with aircraft possibly maneuvering	2005
ASARP [30]	25,600 EU	Encounters with risk of NMAC	2006
MIT correlated [31]	406,728 US	Last moments before NMAC	2008
MIT uncorrelated [8]	NA	Few minutes before collision	2013

This solution may be the most popular because of its realism and its capacity to generate as many encounters as needed. Some popular probabilistic encounter models usable for RPAS include [31], [8]. It can even be integrated in the DAA process, indeed the authors of [33] proposed to use the probabilistic model obtained with such method directly in a DAA algorithm to estimate the risk associated with a maneuver in real time.

The last encounter model, and maybe the most realistic, is a mix between real traffic data and simulated RPAS data. The idea is to define RPAS missions over a given territory and simulate the RPAS flight during these missions. Then, recorded traffic is overlaid on the given territory. By replaying the recorded traffic and making the RPAS follow their missions, encounters are generated. This method has been used in [13], with socio-economical studies defining eight mission types over the whole U.S. territory. It has also been used in the ODREA project [11] with missions defined in the vicinity of an airport to observe encounters between low-level RPAS and arriving/departing aircraft.

Depending on the type of traffic being considered, two types of encounters are possible: correlated and uncorrelated. In correlated encounters, the aircraft involved are cooperative and at least one of them is in contact with ATC. This implies that their motions are linked to some extent through the intervention of ATC. In uncorrelated encounters, at least one of the aircraft is non-cooperative, i.e. not providing its position. The type of encounter impacts the aircraft trajectories as well as the RPAS sensor suite. Indeed, correlated encounters resemble IFR/IFR flights encounters while uncorrelated encounters resemble IFR/VFR flights encounters. For detailed sensors simulation, in correlated encounters the RPAS needs only cooperative sensors while for uncorrelated encounters the RPAS requires cooperative and non-cooperative sensors.

IV. Existing Simulation Frameworks

The previous Section listed the required elements to perform an FTS for CAS performance evaluation. However, for these elements to live, synchronize and communicate together, a simulation framework is needed. In the following, we introduce three freely available simulation frameworks to perform FTS: InCAS, from EUROCONTROL; ACES, from NASA Ames; EASY, from DSNA.

A. InCAS (to become CAFE)

Developed by EUROCONTROL, the InCAS framework [34] allows replaying traffic and analyzing it for CAS performance estimation. It allows analyzing close encounters individually for incident investigations. InCAS processes radar data to provide clean trajectories with a one-second interval. A TCAS module allows simulating collision avoidance logic in the manned airplanes.

A new framework is currently under development, the Collision Avoidance Fast-time Evaluation (CAFE), for airborne and ground-based applications. It will include a close encounter model of the European airspace and rely on rare events simulation methods. Its develoment is being supported by both EUROCONTROL and EUROCAE WG-75, along with SESAR-JU projects.

B. ACES

The Airspace Concept Evaluation System (ACES) is a gate-to-gate simulation architecture developed by NASA Ames [35]. It can replay recorded traffic at a National Airspace System (NAS) or local scale and simulate airspace, aircraft and wind behavior. Its agent based modeling approach provide an independent description for each actor of the airspace. Communications between all agents are available and can be analyzed. A large library of JAVA modules can be plugged on top of the core ones, and custom modules can be added. The execution is sped up by its capability to distribute multiple simulations on a set of machines.

C. EASY

EASY is the simulation framework developed and used by the DSNA, the French Air Navigation Service Provider (ANSP) [36]. It can replay traffic at an international scale with trajectories computed with an adjustable timescale which can go under the second (e.g. for high update rate sensors). Its modular architecture allows plugging-in new modules regardless of the language used to develop them and without needing to access their code (black box modules). A library of existing modules (e.g. pilot behavior, TCAS logic) allows quickly building a full simulation environment. Finally, "as fast as possible" logics and parallelization on multiple machines provide quick computation for large FTS.

This list is not exhaustive, more simulation frameworks can be found in the literature and in the commerce (e.g. CASSAT, SIMMOD, AirTOp, Flames, etc.).

V. Verification and Validation

Based on the FTS outputs, through different metrics, verification and validation give objective marks to the CAS being evaluated. According to the E-OCVM3 validation methodology [37], Verification deals with answering the question "Are we building the system right?". On the other hand, Validation deals with "Are we building the right system?".

A. Verification

A comprehensive way to verify a system is to use formal verification methods. By modelling a system and checking that this model always respects given rules it is possible to ensure that the system will always behave as expected. Though highly effective, these methods require a lot of work to set up. Moreover, as noted in [38], they ask for the system to follow particular designs. Example of such approaches are available in [39], [40], [41], [42].

A more empirical, nevertheless effective, way to verify a CAS is to simulate multiple encounters and verify if the system behaves as expected. This kind of approach requires a large number of encounters in order to explore as many as possible of the situations that could arise. In [32], the authors propose to use evolutionary computation to create encounters from a given encounter model and search for difficult cases to explore from the simplest to the hardest cases.

B. Validation

Various actors are impacted by the behavior of a CAS: the public, RPAS operators, remote pilots, pilots, ATM operators, each one with different expectations. This is reflected through different metrics used to validate a CAS. According to Holland et al. [43], these metrics can be divided into three categories: safety, operational and acceptability metrics.

Safety metrics are directly related to the capacity of the CAS to prevent NMACs. Operational metrics measure the disruption of the avoidance maneuver on the different airspace actors, including ATC and the other aircraft. Acceptability metrics relate to the remote pilot expectations. These metrics reflect how much a remote pilot can trust the system to act as the pilot would have.

The most common safety metric is the risk ratio; it compares the probability of NMAC with and without the CAS. It does not reflect the intrinsic safety of the CAS but rather the safety gains compared to no/another system. In facts, the risk ratio is composed of two quantities: the risk reduction from the CAS and the risk induced by the CAS. Indeed, the CAS may alert when not necessary and the resulting maneuver may increase the risk compared to no CAS. In the case of RPAS, it has been noted in [44] that, due to their lower climbing rate, encounters range rate wherer lower and so was the risk ratio. In order to factor out this element, a modified risk ratio is proposed which consider the RPAS as having climb rates similar to the rest of the traffic when computing the denominator of the risk ratio. Table III lists some common validation metrics. More details can be found in [45], [43], [18], [46], [12], [11].

VI. Conclusions

Often regarded as a cumbersome evaluation method, FTS can be set up quickly thanks to existing tools and results. With the proper validation metrics, they provide rich insight in the pros and cons of a CAS. This paper gathered multiple results and references to allow the reader to quickly start using FTS and adapt it to its needs. Future work aims at creating a benchmark widely available with a plug-and-play feature to allow CAS developers to compare their system on an identical setup.

Acknowledgment

This work was supported by the ENGIE Ineo - Groupe ADP - Safran RPAS Chair. The authors would like to thank Catherine Ronflé-Nadaud, Julien Farjon, Eric Blond and Christian Aveneau for sharing their extensive knowledge on the topic.

References

[1] SESAR-JU. [Online]. Available: http://www.sesarju.eu/sites/default/files/documents/reports/European_Drones_Outlook_Study_2016.pdf

[2] A. R. Lacher, D. R. Maroney, and A. D. Zeitlin, "Unmanned aircraft collision avoidance–technology assessment and evaluation methods," in *FAA EUROCONTROL ATM R&D Symposium, Barcelona, Spain*, 2007.

[3] M. Orefice and V. Di Vito, "Aircraft automatic collision avoidance using spiral geometric approach," *World Academy of Science, Engineering and Technology, International Journal of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, vol. 10, no. 4, pp. 654–661, 2016.

[4] L. R. Sahawneh, M. E. Argyle, and R. W. Beard, "3d path planning for small uas operating in low-altitude airspace," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2016, pp. 413–419.

[5] M. Radmanesh and M. Kumar, "Grey wolf optimization based sense and avoid algorithm for uav path planning in uncertain environment using a bayesian framework," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2016, pp. 68–76.

[6] Y. Lin and S. Saripalli, "Sampling based collision avoidance for uavs," in *2016 American Control Conference (ACC)*. IEEE, 2016, pp. 1353–1358.

[7] R. E. Weibel, M. W. Edwards, and C. S. Fernandes, "Establishing a risk-based separation standard for unmanned aircraft self separation," in *Proceedings of the ninth USA/Europe Air Traffic Management Research & Development Seminar. Berlin, Germany: Eurocontrol/FAA*, 2011.

[8] A. J. Weinert, E. P. Harkleroad, J. Griffith, M. W. Edwards, and M. J. Kochenderfer, "Uncorrelated encounter model of the national airspace system, version 2.0," DTIC Document, Tech. Rep., 2013.

[9] C. Allignol, N. Barnier, N. Durand, G. Manfredi, and E. Blond, "Integration of uas in terminal control area," 2016.

[10] Y. Lyu, Q. Pan, C. Zhao, C. Yu, and J. Hu, "A uav sense and avoid system design method based on software simulation," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2016, pp. 572–579.

[11] ODREA, "Operational demonstration of rpas in european airspace (odrea)," 2015.

[12] MIDCAS, "Mid-air collision avoidance system (midcas)," 2015.

[13] C. Park, S. M. Lee, and E. R. Mueller, "Investigating detect-and-avoid surveillance performance for unmanned aircraft systems," in *Proceedings of 14th AIAA Aviation Technology, Integration, and Operations Conference, AIAA*, vol. 2413, 2014.

[14] J. R. Comstock Jr, R. W. Ghatas, M. J. Vincent, M. C. Consiglio, C. Muñoz, J. P. Chamberlain, P. Volk, and K. E. Arthur, "Unmanned aircraft systems human-in-the-loop controller and pilot acceptability study: Collision avoidance, self-separation, and alerting times (cassat)," 2016.

[15] J. R. Murphy, P. S. Hayes, S. K. Kim, W. Bridges, and M. Marston, "Flight test overview for uas integration in the nas project," in *AIAA Atmospheric Flight Mechanics Conference, AIAA SciTech,(AIAA 2016-1756)*, 2016.

[16] T. Arino, K. Carpenter, S. Chabert, H. Hutchinson, T. Miquel, B. Raynaud, K. Rigotti, and E. Vallauri, "Studies on the safety of acas ii in europe," *Eurocontrol, Technical Rep. ACASA/WP-1.8 D*, 2002.

[17] B. Raynaud, T. Arino, C. Aveneau, and G. Dean, "Applicability and usefulness of the encounter model-based methodology for safety net assessment."

[18] S. P. Cook, D. Brooks, R. Cole, D. Hackenberg, and V. Raska, "Defining Well Clear for Unmanned Aircraft Systems," in *AIAA Infotech @ Aerospace*. American Institute of Aeronautics and Astronautics, 2015. [Online]. Available: http://arc.aiaa.org/doi/abs/10.2514/6.2015-0481

[19] M. T. DeGarmo, "Issues concerning integration of unmanned aerial vehicles in civil airspace," *The MITRE Corporation Center for Advanced Aviation System Development*, 2004.

[20] M. Camarero Sanchez and L. Cabedo Vidal, "Creation of rpas aircraft performance models for bada." 2015.

TABLE III: Example of validation metrics commonly used for CAS safety assessments.

Safety			
	Mitigated risk ratio	$\frac{NMAC_{CAS}}{NMAC_{noCAS}}$	Denote the safety gain when using the proposed CAS.
	Modified risk ratio	$\frac{NMAC_{CAS}}{NMAC_{noCAS,noClimbLimit}}$	Similar to mitigated risk ratio, but factors out the fact that RPAS climb slowly.
	Relative risk ratio	$\frac{NMAC_{CAS1}}{NMAC_{CAS2}}$	Denote the safety gain when using CAS1 instead of CAS2.
	Unresolved risk ratio	$\frac{NMAC_{unresolved}}{NMAC_{noCAS}}$	This number reflects the proportion of encounters unresolved by the CAS. Risk ratio = unresolved risk ratio + induced risk ratio.
	Induced risk ratio	$\frac{NMAC_{induced}}{NMAC_{noCAS}}$	This number reflects the proportion of encounters created by maneuvers provided by the CAS. Risk ratio = unresolved risk ratio + induced risk ratio.
	Positive predictive value	$\frac{TP}{TP+FP}$	The probability of declaring an alert when necessary relative to the total number of alerts.
	False discovery rate	$\frac{FP}{TP+FP}$	The probability of declaring an alert when unnecessary relative to the total number of alerts.
	True Positive (TP) rate	$\frac{TP}{TP+FN}$	The probability of declaring an alert when a collision risk exists.
	False Positive (FP) rate	$\frac{FP}{FP+TN}$	The probability of declaring an alert when no collision risk exists.
	Missed Alert rate	$\frac{FN}{TP+FN}$	The probability of not declaring an alert when a collision risk exists.
	Vertical miss distance at CPA	Δh_{CPA}	Provides an insight on how close the avoidance takes place and if the resulting maneuver has been successful.
	Horizontal miss distance at CPA	Δh_{CPA}	Provides an insight on how close the avoidance takes place and if the resulting maneuver has been successful.

[21] E. Theunissen, B. Suarez, and F. Kunzi, "Well clear recovery for detect and avoid," in *Digital Avionics Systems Conference (DASC), 2016 IEEE/AIAA 35th*. IEEE, 2016, pp. 1–10.

[22] Eurocontrol, *Airborne Collision Avoidance Systems (incorporating TCAS II version 7.0 & 7.1 and introduction to ACAS X)*. Eurocontrol, 2015.

[23] M. J. Kochenderfer, J. E. Holland, and J. P. Chryssanthacopoulos, "Next generation airborne collision avoidance system," *Lincoln Laboratory Journal*, vol. 19, no. 1, pp. 17–33, 2012.

[24] N. Musavi, D. Onural, K. Gunes, and Y. Yildiz, "Unmanned aircraft systems airspace integration: A game theoretical framework for concept evaluations," *Journal of Guidance, Control, and Dynamics*, pp. 96–109, 2016.

[25] S. C. Mohleji and G. Wang, "Modeling ads-b position and velocity errors for airborne merging and spacing in interval management application," *MITRE release*, pp. 10–3026, 2010.

[26] Vega Prime. [Online]. Available: http://www.presagis.com/products_services/products/modeling-simulation/visualization/vega_prime/

[27] J. Yu, D. Tang, X. Chen, and L. Hu, "Application of dynamic ir scene generation technique in airborne electro-optical detecting simulation systems," in *Electronic Measurement Instruments, 2009. ICEMI '09. 9th International Conference on*, Aug 2009, pp. 3–840–3–843.

[28] A. Mcfadyen and L. Mejias, "A survey of autonomous vision-based see and avoid for unmanned aircraft systems," *Progress in Aerospace Sciences*, vol. 80, pp. 1–17, 2016.

[29] B. Raynaud and Arino, "Implications on acas performances due to asas implementation," *Eurocontrol, Technical Rep. IAP A/WP11/114/D*, 2005.

[30] ——, "Asarp: Acas safety analysis post-rvsm project," *Eurocontrol, Technical Rep. ASARP/WP9/72/D*, 2006.

[31] M. Kochenderfer, L. Espindle, J. Kuchar, and J. D. Griffith, "Correlated encounter model for cooperative aircraft in the national airspace system version 1.0," Tech. Rep., 2008.

[32] X. Zou, R. D. Alexander, and J. A. McDermid, "On the validation of a uav collision avoidance system developed by model-based optimization:: Challenges and a tentative partial solution," 2016.

[33] J. K. Kuchar and L. C. Yang, "A review of conflict detection and resolution modeling methods," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 4, pp. 179–189, Dec. 2000.

[34] EUROCONTROL. [Online]. Available: http://www.eurocontrol.int/articles/acas-ii-monitoring-interactive-collision-avoidance-simulator-incas

[35] S. George, G. Satapathy, V. Manikonda, K. Palopo, L. Meyn, T. Lauderdale, M. Downs, M. Refai, and R. Dupee, "Build 8 of the airspace concept evaluation system," in *AIAA modeling and simulation technologies conference*, 2011, p. 6373.

[36] EUROCONTROL. [Online]. Available: http://www.eei.cena.fr/products/easy/easy_leaflet.pdf

[37] EUROCONTROL, "E-ocvm version 3.0 volume i: European operational concept validation methodology," EUROCONTROL, Tech. Rep., 2010.

[38] G. Hagen, R. Butler, and J. Maddalon, "Stratway: A modular approach to strategic conflict resolution," in *Preceedings of 11th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference, Virgina Beach, VA*, 2011.

[39] J.-B. Jeannin, K. Ghorbal, Y. Kouskoulas, R. Gardner, A. Schmidt, E. Zawadzki, and A. Platzer, "A formally verified hybrid system for the next-generation airborne collision avoidance system," in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2015, pp. 21–36.

[40] C. Muñoz, A. Narkawicz, G. Hagen, J. Upchurch, A. Dutle, M. Consiglio, and J. Chamberlain, "Daidalus: detect and avoid alerting logic for unmanned systems," in *2015 IEEE/AIAA 34th Digital Avionics Systems Conference (DASC)*. IEEE, 2015, pp. 5A1–1.

[41] C. A. Muñoz, A. Dutle, A. Narkawicz, and J. Upchurch, "Unmanned aircraft systems in the national airspace system: a formal methods perspective," *ACM SIGLOG News*, vol. 3, no. 3, pp. 67–76, 2016.

[42] C. von Essen and D. Giannakopoulou, "Probabilistic verification and synthesis of the next generation airborne collision avoidance system," *International Journal on Software Tools for Technology Transfer*, vol. 18, no. 2, pp. 227–243, 2016.

[43] J. E. Holland, M. J. Kochenderfer, and W. A. Olson, "Optimizing the next generation collision avoidance system for safe, suitable, and acceptable operational performance," *Air Traffic Control Quarterly*, vol. 36, 2014.

[44] E. H. Londner, "Collision avoidance system effectiveness on low performance unmanned aircraft," in *AIAA Infotech@ Aerospace*, 2016, p. 1987.

[45] S. M. Lee, C. Park, D. P. Thipphavong, D. R. Isaacson, and C. Santiago, "Evaluating alerting and guidance performance of a uas detect-and-avoid system," 2016.

[46] L. Espindle, J. Griffith, and J. Kuchar, "Safety analysis of upgrading to tcas version 7.1 using the 2008 us correlated encounter model," *Project Report ATC-349, Lincoln Laboratory, Lexington, Mass*, 2009.