DEVELOPMENT OF A HIGH-PRECISION ADS-B BASED CONFLICT ALERTING SYSTEM FOR OPERATIONS IN THE AIRPORT ENVIRONMENT

Fabrice Kunzi and R. John Hansman

This report is based on the Doctoral Dissertation of Fabrice Kunzi submitted to the Department of Aeronautics and Astronautics in partial fulfillment of the requirements for the degree of Doctor of Philosophy at the Massachusetts Institute of Technology. The work presented in this report was also conducted in collaboration with the members of the Doctoral Committee:
Prof. Hamsa Balakrishnan, Dr. Jim Kuchar, Dr. Tom Reynolds

Report No. ICAT-2013-09
October 2013

MIT International Center for Air Transportation (ICAT)
Department of Aeronautics & Astronautics
Massachusetts Institute of Technology
Cambridge, MA 02139 USA

DEVELOPMENT OF A HIGH-PRECISION ADS-B BASED CONFLICT ALERTING SYSTEM FOR OPERATIONS IN THE AIRPORT ENVIRONMENT

by

Fabrice Kunzi

ABSTRACT

The introduction of Automatic Dependent Surveillance - Broadcast (ADS-B) as the future source of aircraft surveillance worldwide provides an opportunity to introduce high-precision airborne conflict alerting systems for operations in high-density traffic environments. Current alerting systems have been very successful at preventing mid-air collisions in the en-route environment but have limited benefit in high-density environments such as near airports where most mid-air collisions occur (59%). Furthermore, introducing an ADS-B–enabled conflict alerting system generates an incentive for General Aviation users to voluntarily equip with ADS-B avionics.

The work presented in this thesis describes the process followed to develop an ADS-B–enabled, high-precision conflict alerting system. This system will be the basis for the international certification standard guiding future implementations of such systems. The work was conducted as part of the larger development effort of the Traffic Situation Awareness with Alerting (TSAA) ADS-B application.

As a first step, a set of 18 high-level system requirements was identified based on a stakeholder analysis and review of mid-air collisions that occurred over the last 10 years. An alerting algorithm was then developed based on the system requirements that builds on the precedent set by current alerting systems but takes advantage of the improved state information available via ADS-B. The distinguishing factors of the algorithm are its use of a constant turn rate trajectory prediction and its consideration of the current and predicted encounter geometry in the alerting decision.

Next, a method to tune the performance of the algorithm was developed and demonstrated. The method applies the Latin hypercube sampling approach to generate a large set of different algorithm implementations, which were then evaluated by simulating the alerting performance on a representative data set of airborne encounters. Lastly, the method introduced an approach to evaluate and "visualize" the five-dimensional performance space defined by the five performance metrics of interest for alerting systems.

Using the tuned algorithm, a flight test program was conducted. The performance of the algorithm during the flight test was analyzed in-depth and compared to the expected performance. Given the insights from the tuning and the flight test, additional alerting logic was introduced to the basic algorithm, which significantly improved overall alerting performance.

The performance of the final system implementation is significantly better or equal to that of the current industry standard for all five performance metrics. The nuisance and overall alert rate were each reduced by a factor of more than 4 and the average time of alert before the closest point of approach increased by 6 seconds as compared to current systems. Enabled by this performance improvement, TSAA introduces reliable collision alerting to the Airport Environment where most of today's mid-air collisions occur and where today's alerting systems are of limited benefit due to high rates of nuisance alerts.

Thesis Supervisor: R. John Hansman, Professor of Aeronautics and Astronautics

ACKNOWLEDGMENTS

Many highly competent people that cannot go unmentioned supported the work presented in this thesis. It is in large part because of their efforts that the project was a success.

At MIT, the mentoring, guidance and support of John Hansman has been invaluable to the project as well as to me personally. I am very grateful for everything that he has done for me. Also at MIT, Sathya Silva and HongSeok Cho were a tremendous team for conducting the Human Factors evaluations for TSAA. During the initial phase of the project, Maxime Gariel provided great guidance for the algorithm development. Thanks are also in order for the members of my PhD thesis committee – Hamsa Balakrishnan, Jim Kuchar and Tom Reynolds – who brought very helpful perspectives to my work, significantly improving the final quality of this thesis.

At the FAA, the skillful management of the entire TSAA project by David Gray was the reason the project was able to achieve an aggressive schedule and deliver a usable product. Additionally, the input and support of Doug Arbuckle and Don Walker was always beneficial. It is great to know that the FAA has such highly capable individuals leading the upgrade of the national airspace system under NextGen. It was the FAA's Surveillance and Broadcast Services Office that funded this project under contract number Z988401.

At MITRE, Dave Elliott, Doug Havens and Kara MacWilliams were instrumental during the performance and safety analysis of TSAA. They helped share the load of all the analysis that was required for TSAA; a task that I could not have achieved by myself.

At Avidyne, the project received significant support from Ted Lester, Dean Ryan, Duane Ott, Mike Keirnan and Trevor Steffensen.

Additional help came from Mykel Kochenderfer who served as a reader and taught me the correct way to think about uncertainty, from Jim Duke who brought the commercial pilot's view to the table, and by the member of RTCA Special Committee SC-186's WG4, who represented the interest of the international community during the standards development.

Lastly, but most importantly, my wife Alyssa is the reason that I didn't loose my mind over the last years. I feel very blessed and honored to have her by my side, ready and willing to help me think through tough engineering problems, edit papers and presentations and pick me up at the lab at ungodly hours of the night. You are incredible – thank you for everything you do for me.

TABLE OF CONTENTS

LIST OF FIGURES

Number *Page*

LIST OF TABLES

Number *Page*

Chapter 1

INTRODUCTION AND MOTIVATION

Mid-air collisions must be prevented during flight operations. Between 2000 and 2010, 112 mid-air collisions occurred in the United States, 66 (59%) of which occurred in the airport pattern or the immediate vicinity of an airport [1]. Current airborne traffic alerting systems such as the Traffic Alert and Collision Avoidance System (TCAS), originally developed in the 1980s for commercial aviation, have been very successful in preventing mid-air collisions in the en-route environment.

As shown Figure 1-1, TCAS systems are required on all aircraft carrying more than 10 passengers or that have a maximum takeoff weight (MTOW) of 15,000 kg or more. TCAS I systems only alert the flight crew to potential threats, while TCAS II systems also issue executive commands on how to avoid the threat. The Traffic Advisor System (TAS) is a TCAS I implementation specific to General Aviation (GA).

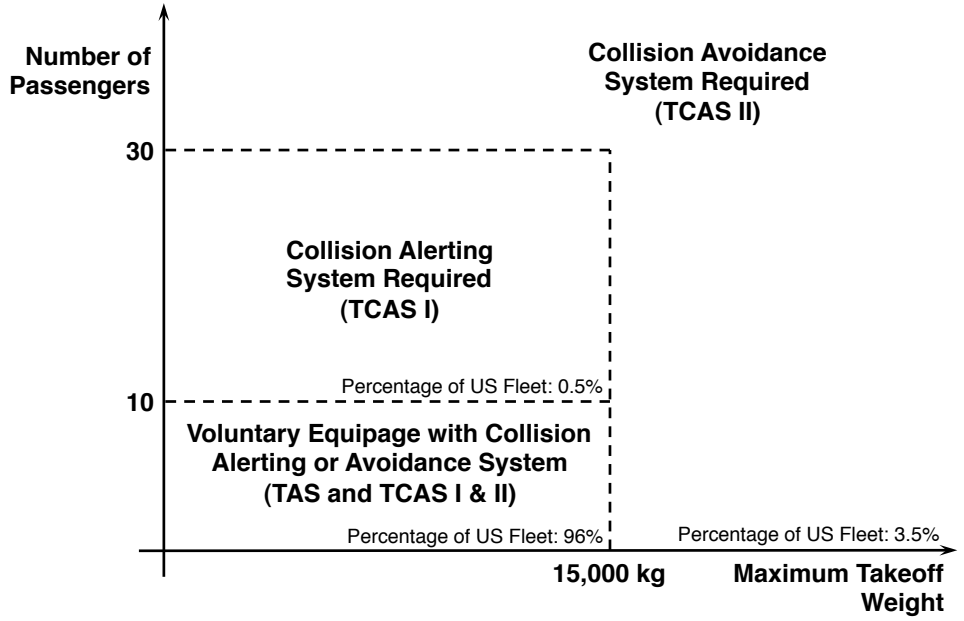


Figure 1-1: Required Traffic Alerting and Avoidance Systems

Due in part to sensor limitations, TCAS I and II systems tend to over-alert when operating in high-density environments such as in the vicinity of an airport [2]. Additionally, in part due to their high cost, voluntary equipage with such alerting systems among aircraft with less than 10 passengers and an MTOW of less than 15,000kg in the United States was at only 14.5% in 2010 [3]. If equipage costs were lower, the increased frequency congestion caused by the systems' active surveillance sensors make high levels of fleet-wide equipage undesirable [4].

In recent years, Automatic Dependent Surveillance–Broadcast (ADS-B) has been introduced worldwide as a new source of aircraft surveillance information. Aircraft equipped with ADS-B transmit more frequent and more comprehensive aircraft surveillance information than what current ground based radar can determine. For ADS-B to function, however, aircraft first must be equipped with ADS-B avionics. To achieve a high level of fleet-wide equipage, airworthiness authorities worldwide have introduced equipage mandates that require aircraft to transmit ADS-B in busy airspace. This mandate takes effect by 2020 in the US and by 2017 in Europe.

Airworthiness authorities and industry are interested in stimulating voluntary equipage of ADS-B avionics across all stakeholders ahead of the mandate, including in airspace where the transmission of ADS-B messages will not required by law [4–7]. One way to stimulate this voluntary equipage is to provide the involved stakeholders with benefits that result from use of the technology ("user benefit"). The more user benefit a stakeholder perceives from a given technology, the more likely that stakeholder is to equip with that technology.

The information transmitted via ADS-B is more comprehensive and has the potential to contain significantly less error than the information available from radars or the collision alerting system sensors mentioned above. With this improved information, ADS-B represents an opportunity to introduce new, high-precision alerting systems that can operate in high-density environments without generating high rates of undesirable alerts. Previous work has identified that introducing ADS-B–enabled conflict alerting to the National Airspace System (NAS) has the potential to generate significant user benefit; thus creating an incentive for stakeholders to equip ADS-B avionics voluntarily [8], [9].

In light of this, MIT has developed a prototype of an ADS-B–enabled airborne conflict alerting system ("exemplar system"). Known as the Traffic Situation Awareness with Alerting (TSAA) ADS-B Application, this exemplar system is to serve as the basis for the international certification standard that will be used to certify such systems in the future. This thesis describes the development of this system.

1.1 The Systems Engineering Approach to the Development TSAA

The development of TSAA followed a standard systems engineering approach. Commonly represented by the V-Model, the process is shown in Figure 1-2. The two components of the V -model are the system or program definition (the downward arrow in Figure 1-2) and the system integration, testing, and operation (the upward arrow in Figure 1-2).

Developing TSAA started with identifying ADS-B as a technological opportunity and combining this with a National Airspace System (NAS) stakeholder assessment in order to outline a high-level system concept for an ADS-B–enabled conflict alerting system. Next, system requirements were defined for the alerting system; and based on those requirements, a prototype alerting system was designed. This alerting system then was evaluated in depth and tuned to the desired performance.

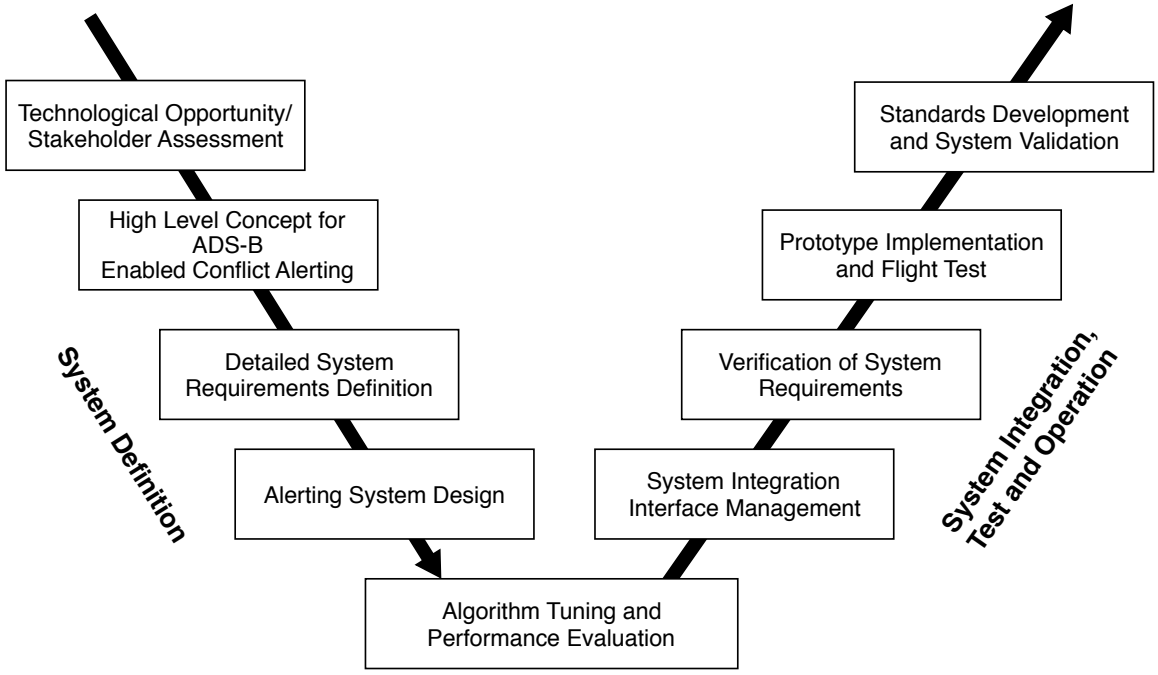


Figure 1-2: The V-Model Systems Engineering Framework Adapted for TSAA

One the upward arrow of Figure 1-2, building a physical system and conducting a set of human factors evaluations and technical studies solidified the physical implementation and architecture of the TSAA system. Once implemented, the system was verified to meet the system requirements set out during the design process through an extensive flight test program. Lastly, the standard for future implementations of TSAA was written.

One important observation about the Systems Engineering V-Model is that each step on the downward arrow of Figure 1-2 maps to a corresponding step on the upward arrow. For example, the stakeholder needs identified during the stakeholder assessment maps to the validation step that assesses whether the system actually meets those needs.

1.2 Thesis Outline and Overview

Figure 1-3 shows the process used to develop TSAA and how it relates to the organization of this thesis. The individual steps loosely map to the systems engineering approach described in section 1.1. An overview of each chapter is provided below.

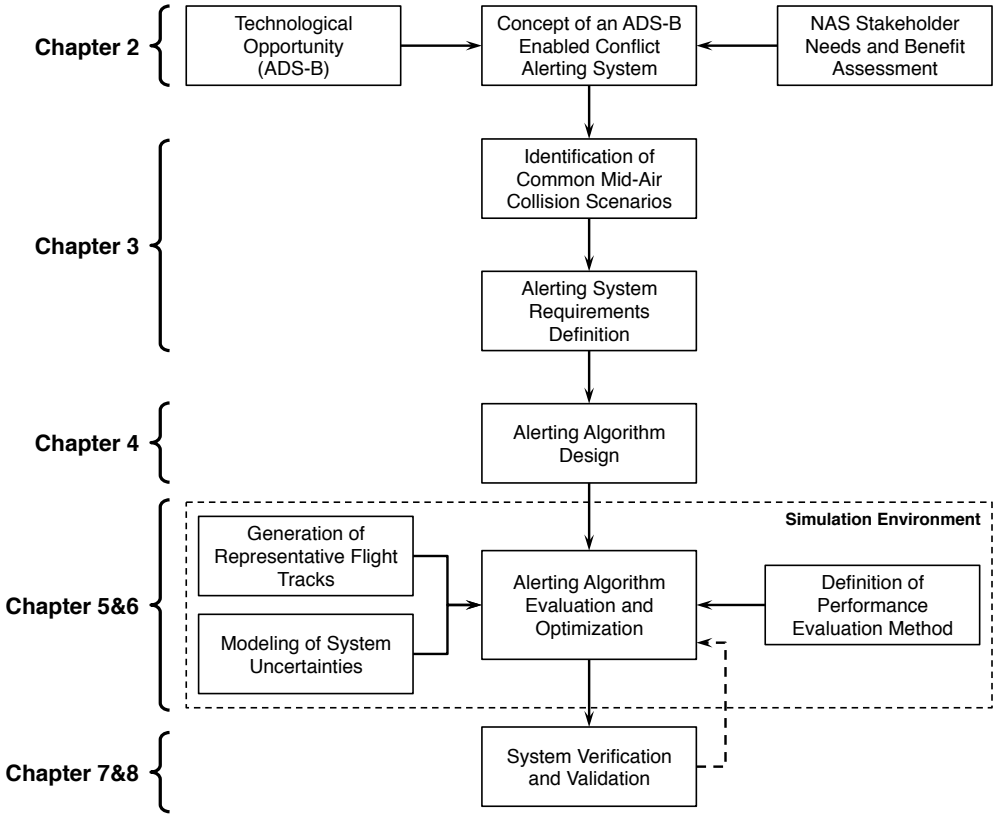


Figure 1-3: Major Thesis Components and Their Relationship

As mentioned above, the surveillance infrastructure upgrade to ADS-B presents an opportunity to introduce high-performance alerting systems as avionics to the NAS. In turn, ADS-B–based alerting systems both incentivize airspace users to equip ADS-B avionics and also introduce a system-wide safety benefit. Chapter 2 reviews current alerting systems and

their advantages and disadvantages in order to evaluate whether existing systems or elements of them can be repurposed for TSAA. Chapter 2 also reviews other research in the field of conflict alerting systems and algorithms.

Chapter 3 defines the system's requirements for the TSAA prototype alerting system. The requirements are based on stakeholder expectations, the literature review conducted in Chapter 2, and an analysis of 10 years' worth of NTSB mid-air collision data. In order to meet the identified system requirements, the decision to design a new algorithm for TSAA was made. Chapter 4 describes this new algorithm and provides a detailed description of its components and implementation.

Chapter 5 introduces a method for the tuning and evaluation of the new TSAA algorithm. Chapter 6 demonstrates the use of this method to obtain the desired algorithm performance to meet the system requirements.

Chapter 7 summarizes the verification and validation efforts that were performed on the overall TSAA system. Specifically, TSAA was evaluated over 3 months of flight tests: the alerting behavior observed during the flight test was compared to the alerting behavior that would be expected from the simulation. Based on the data and insights from the flight tests, additional improvements could be introduced in Chapter 8 to the basic algorithm (signified by the dashed feedback path in Figure 1-3), significantly improving its performance.

Chapter 9 summarizes the major points and components of this thesis and identifies areas of further work.

Chapter 2

BACKGROUND AND LITERATURE REVIEW

The fundamental task of a collision alerting system is to decide whether an alert must be issued to the flight crew, given some information about the environment surrounding its own aircraft ("own-ship"). This chapter provides a general description and representation of this task, reviews how it is implemented in current airborne alerting systems, and summarizes other approaches that have been proposed in literature.

2.1 General Representation of the Alerting Problem and Alerting Systems

Collision alerting systems have been studied in depth as part of a larger research effort on hazard alerting systems. In the larger context of hazard alerting systems, two types of alerting systems can be identified. The first is a *reactive* alerting system, which alerts to the observed presence of a hazard. An example of a reactive alerting system would be a system that alerts to the presence of an engine fire. The second is a *predictive* alerting system, which alerts when a hazard is predicted to be present in the future. An example of a predictive alerting system would be a conflict alerting system that alerts to the possibility of a mid-air collision in the future.

Kuchar provides an in-depth discussion on alerting systems and generalizes the alerting task in a state-space representation. This state-space representation allows for the analysis of specific issues affecting alerting systems while retaining generality across many applications; it is shown in Figure 2-1 [10]. Notional states x_1 and x_2 are the input states and together define the Alerting State Space X. The time evolution of those states define the state trajectory vector **x(t)**. The alert region represents the ranges of the states x_1 and x_2 that are considered to indicate that the hazard against which the system protects is present in the system. It should be noted that the alert region is not synonymous with the hazard itself, but rather defines the state-space region that would be considered hazardous.

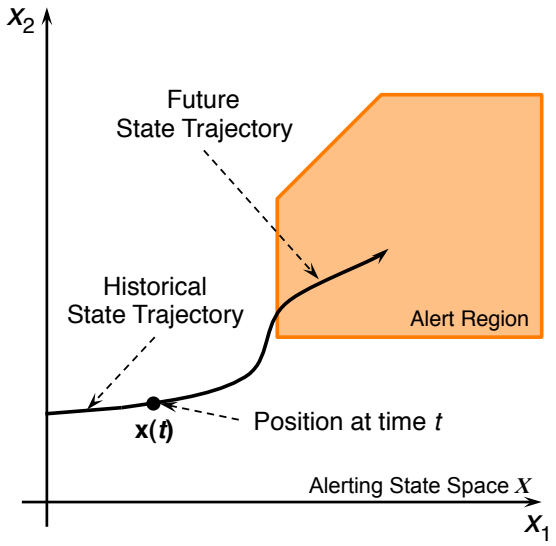


Figure 2-1: State-Space Representation of the Alerting Problem

Using the state space representation of a hazard alerting system, the two alerting system categories can be described more precisely, as shown in Figure 2-2.

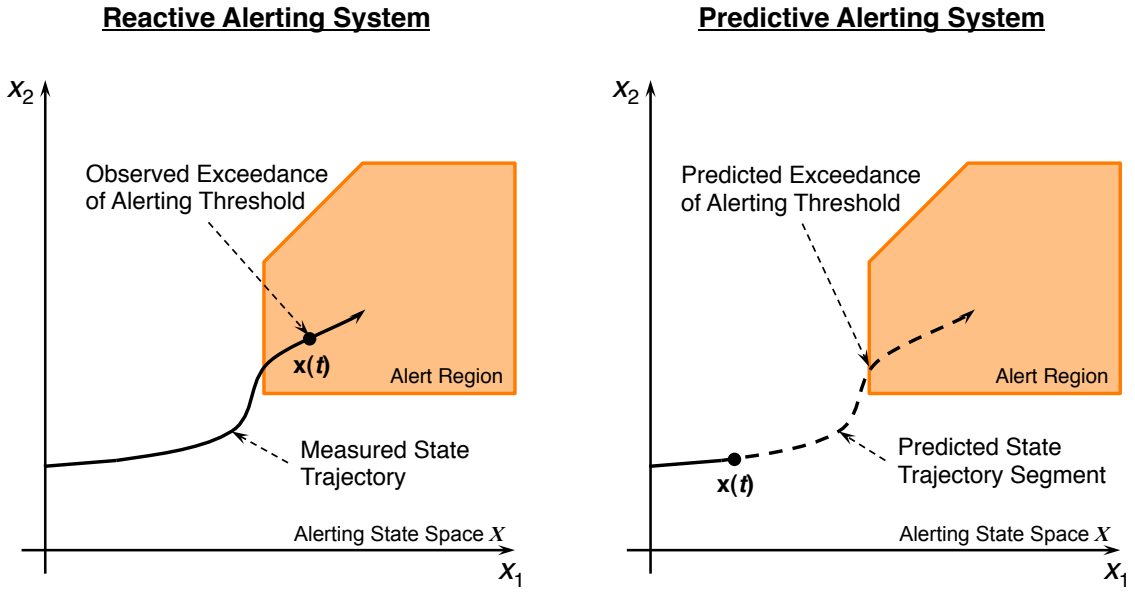


Figure 2-2: Schematic Representation of a Reactive Alerting System (left) and Predictive Alerting System (right)

For a reactive alerting system, shown on the left in Figure 2-2, an alert is issued if the current states are within the alert region. In a predictive alerting system, as shown on the right, the system projects a state trajectory segment into the future and an alert is issued if this segment penetrates the alert region.

Most commonly, airborne alerting systems are predictive alerting systems and predict how the states currently known will evolve along a trajectory segment. If a set of conditions are met along this predicted trajectory segment, an alert is issued. Most of the time, however, neither the current states nor their evolution over time is known with absolute certainty. The next section discusses how this limitation of knowledge affects predictive alerting systems.

2.2 Uncertainty in Predictive Alerting Systems

A schematic representation of the functions and information flow for predictive alerting systems is shown in Figure 2-3. As the operations of interest occur in the surrounding environment, a sensor measures the states required by the alerting system. Based on those states, the alerting system decides whether a hazard is present and whether an alert to the operator is necessary. If an alert is issued, the flight crew then decides how to respond to the alert, which in turn affects the operations in the original environment.

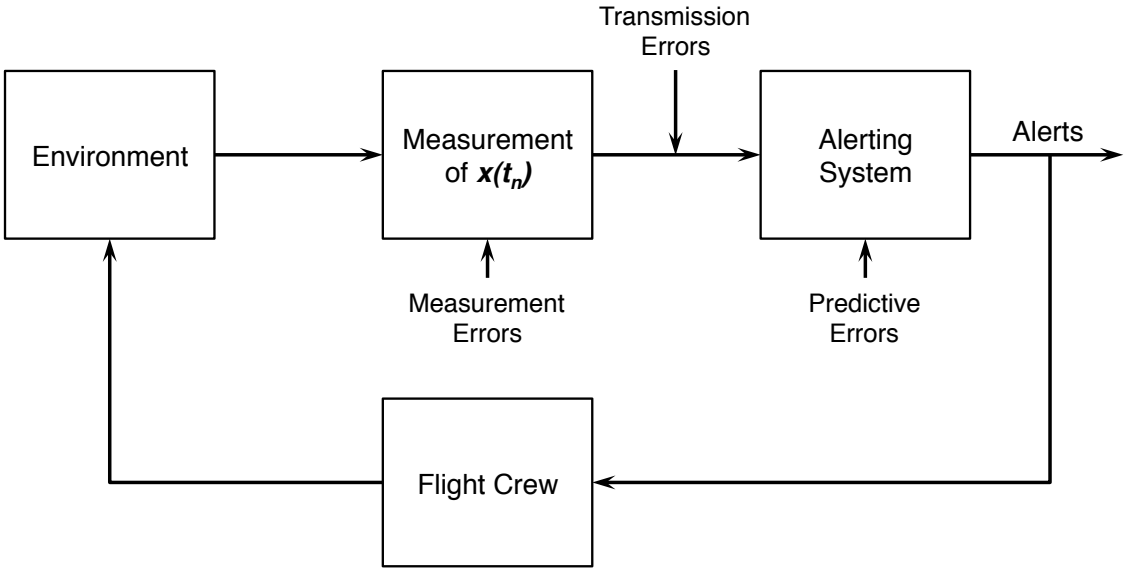


Figure 2-3: Schematic Representation of Functions and Information Flow in a Conflict Alerting System

During the process of measuring the states, sensor limitations introduce state errors to the measured information. Additional errors are introduced during the transmission of the measured states, such as errors due to latency compensation. Together, measurement and transmission errors introduce uncertainty about the system's current state. This uncertainty is referred to as "current state uncertainty". Additionally, since future operations are unknown, the state predictions made by alerting systems are inherently uncertain, which introduces predictive errors. This type of uncertainty is referred to as "future state uncertainty". The presence of current and future state uncertainty significantly affects the design of alerting systems and composes much of the literature on alerting system design [11]. Figure 2-4 shows conceptual representations of the errors that generate the future and current state uncertainty.

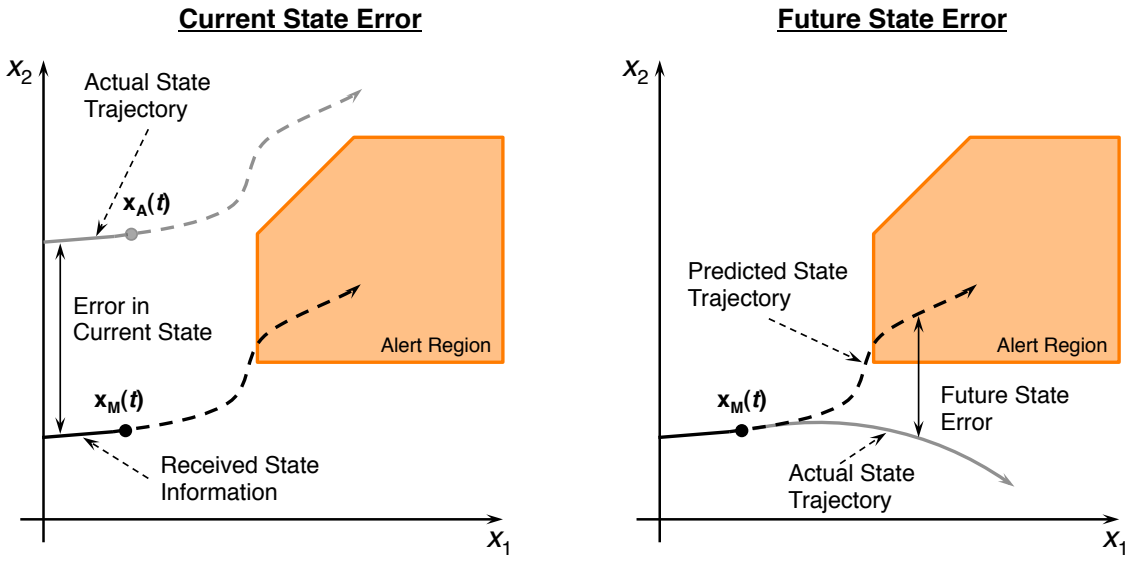


Figure 2-4: State Space Representation of Current State Error (left) and Future State Error (right)

2.2.1 CURRENT STATE UNCERTAINTY

Current state uncertainty—that is, uncertainty associated with the currently known states—is the statistical distribution of the errors between the true state and the measured state, as sampled over repeated measurements. Rowe refers to this type of uncertainty as metrical uncertainty [12]. Beyer and Sendhoff describe state uncertainties as "objective uncertainties" that are of an "intrinsically irreducible stochastic nature" [13].

The stochastic behavior of current state errors (i.e., sporadic, non-deterministic) is frequently complex but often can be described as a combination of a slow moving bias with a superimposed Gaussian jitter [14]. Therefore, the measurement error at any given time

depends on past error on one hand and on random, unpredictable elements on the other. The proportions with which those two components affect the total state error fundamentally depend on the sensor generating the state measurement $\mathbf{x_M(t)}$.

In Figure 2-5, the current state uncertainty is illustrated by an oval around $\mathbf{x_M(t)}$. It should be noted that the dashed line does not imply that the state uncertainty is a discrete ellipse; rather, due to the stochastic nature of state uncertainty, it is generally defined with a probability distribution. As such, the edge of the ellipse can be viewed as a percentage bound of the uncertainty that gives the probability of the measured state falling into that ellipse.

2.2.2 FUTURE STATE UNCERTAINTY

In order to alert to the future presence of a hazard, predictive alerting systems project a future state trajectory segment (represented as a dashed line in Figure 2-2 [right] and Figure 2-5) and evaluate whether the hazard will be present along that trajectory. However, since operator intent is generally unknown, such predictions are inherently uncertain, which introduces future state uncertainty. Formally, future state uncertainty is the distribution of the differences between the predicted and the actual future trajectory, as sampled over repeated predictions [15]. As stands to reason, the more accurately the prediction matches the true future trajectory, the lower the future state uncertainty will be and thus the more likely the system will issue accurate alerts. In Figure 2-4 (right), the alert region would be avoided if the alerting algorithm predicted the actual trajectory more accurately. As is shown in Figure 2-5, the further into the future a system predicts, the greater its future state uncertainty becomes; this relationship defines an "uncertainty cone".

It is important to note that some of the errors that may be present in the current state information have the potential to affect the future state uncertainty significantly. One such error is the velocity error: if velocity is used to predict future states, errors in its direction (i.e., heading or track angle) or magnitude will result in errors about the future states as well. Section 5.5.2 discusses this phenomenon in more detail.

2.2.3 APPROACHES TO REDUCING EFFECTS OF CURRENT AND FUTURE STATE UNCERTAINTY

Various approaches to reduce the effects of uncertainty on alerting systems across transportation methods have been proposed [16] . Generally, the proposals can be grouped into three categories. The first category includes approaches that attempt to reduce the effects of the current state uncertainty, effectively reducing the size of the ellipse around $\mathbf{x_M(t)}$ in Figure 2-5. The literature on stochastic estimation largely overlaps with this category of proposed approaches, and will be discussed in later sections. The second

category comprises attempts to improve the predicted states and thus to reduce future state uncertainty, effectively reducing the width of the cone at $\mathbf{x_M(t+\Delta t)}$ in Figure 2-5. Lastly, approaches in the third category combine knowledge about the current state uncertainty with knowledge about potential future maneuvers and approach the alerting decision from a decision theoretic view, recognizing the fact that uncertainty is present in the alerting decision itself instead of minimizing it ahead of the alerting decision. Thus, the alerting decision is optimized while directly taking that uncertainty into account.

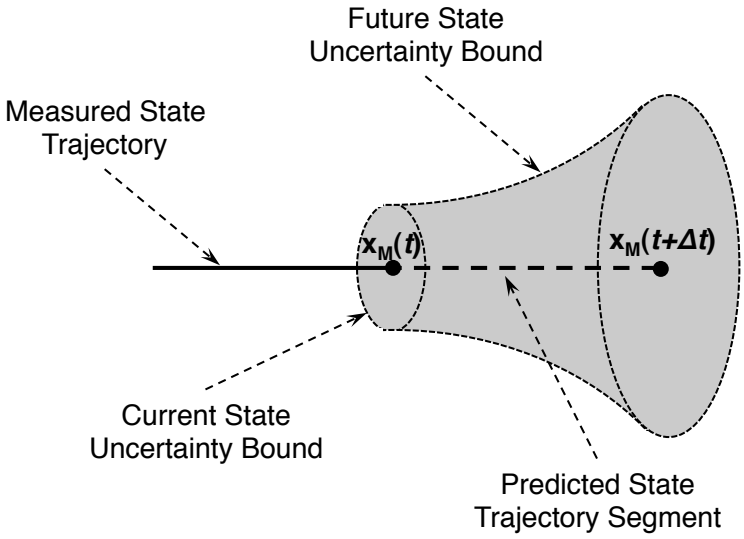


Figure 2-5: Schematic Representation of Combined Current and Future State Uncertainty

The first category focuses on reducing the uncertainty associated with the current state estimate. Across transportation methods, Kalman filters commonly are used to account for and reduce the effects of the current state uncertainty [17], [18]. Certain algorithms of automotive alerting systems attempt to model road conditions directly as well as modeling engine and brake performance in the ambient conditions to predict and therefore avoid collisions [19]. Alterovitz et al. propose using a Markov decision process to take possible future states into account and thus optimize the current alerting decision [20]. More recently, Jansson and Gustafsson proposed a framework for collision-avoidance algorithms that use online Monte Carlo techniques to convert state measurements with stochastic errors to Bayesian risk to evaluate whether an alert should be issued [21]. For the naval industry, a conflict avoidance system based on a genetic algorithm as a means to reduce the threat of environmental pollution due to oil tanker collisions has been developed [22].

Similar advances to address current state uncertainty in airborne alerting systems have also been proposed. However, as observed by Hwang et al., a single filter may not be sufficient to

estimate states in systems with various operational modes (e.g., airport vs. en-route) [23]. In order to detect mode changes, multi-modal Kalman filters or the use of intent information have been proposed [23], [24]. Only recently has the first performance standard using three independent, two-state Kalman filters to account for state uncertainty in airborne target tracking been published; section 3.3 discusses this standard in depth [25].

Of the approaches to reduce the effects of uncertainty that focus on reducing the uncertainty associated with the predicted future states, three types of trajectory predictions are commonly used: discrete, probabilistic, and worst case [16]. Kuchar and Yang proposed a conflict alerting system that uses probabilistic models for state and predictive uncertainties, and estimates the probability of a conflict using Monte Carlo sampling methods [26]. Eby and Kelly proposed an algorithm derived from potential-field models [27] and Chiang and Klosowski proposed using a geometric algorithm for conflict detection and resolution [28]. Building on the probabilistic approach used in the Kuchar and Eby papers, Jones proposed a real-time probabilistic collision avoidance algorithm for autonomous vehicles [29]. Prandini et al. also used a probabilistic framework for trajectory prediction to detect potential future aircraft conflicts [30–32]. At NASA Langley, Munoz et al. developed multiple algorithms that use probabilistically derived buffer zones around the own-ship and the target aircraft [33–39]. Yet another approach, by Christodoulou and Kodaxakis, solved the alerting problem using a mixed-integer problem formulation [40]. In Europe, Eurocontrol standards for trajectory prediction for short-term conflict detection have been published [41–43]. A comprehensive survey of 68 conflict detection and resolution methods up to 2000 is presented by Kuchar and Yang [16]. Erzberger and Paielli propose a model for the error due to trajectory prediction[44].

The last category takes a more holistic view of alerting in the presence of uncertainty and approaches the alerting problem from the perspective of decision theory. As opposed to using thresholds, the alerting decision is made based on the expected utility or value of the alert [45]. If that utility is high enough, the decision to alert is made. The strength of this approach is that alerts are delayed in situations where the current or future state uncertainty is high until the expected utility is high enough, which reduces the number of unnecessary alerts. One approach in this category is proposed by Yang and builds on the probabilistic conflict alerting system proposed by Kuchar and Yang above. The approach uses the expected performance of the alerting system as a decision metric for when to issue an alert [46], [47]. More recently, a significant body of work has been generated on this type of alerting system as part of the development of the Airborne Collision Avoidance System (ACAS) at Lincoln Laboratory, and is addressed in a later section.

2.3 Airborne Collision Alerting Systems Currently in Use

Most current airborne collision alerting systems are designed to meet one of two standards:

- **RTCA/DO-197**, "Minimum Operational Performance Standards for An Active Traffic Alert and Collision Avoidance System I (ACTIVE TCAS I)"
- **RTCA/DO-185**, "Minimum Operational Performance Standards for Traffic Alert and Collision Avoidance System II (TCAS II)"

In the 1980s, in response to a series of mid-air collisions involving commercial aircraft, the US Congress tasked the Federal Aviation Administration (FAA) to develop and mandate the Traffic Alert and Collision Avoidance System (TCAS; public law 100-223). TCAS uses an active sensor on-board the own-ship to interrogate transponder-equipped aircraft in the vicinity to evaluate if they pose a threat. In addition to determining whether an aircraft poses a threat, TCAS II systems have the capability to calculate a maneuver, coordinate it with the other aircraft if it is also equipped with TCAS II, and issue commands directing the flight crew how to execute that maneuver. The conflict alerting and avoidance algorithm developed for TCAS that performs this evaluation is the basis for both standards listed above.

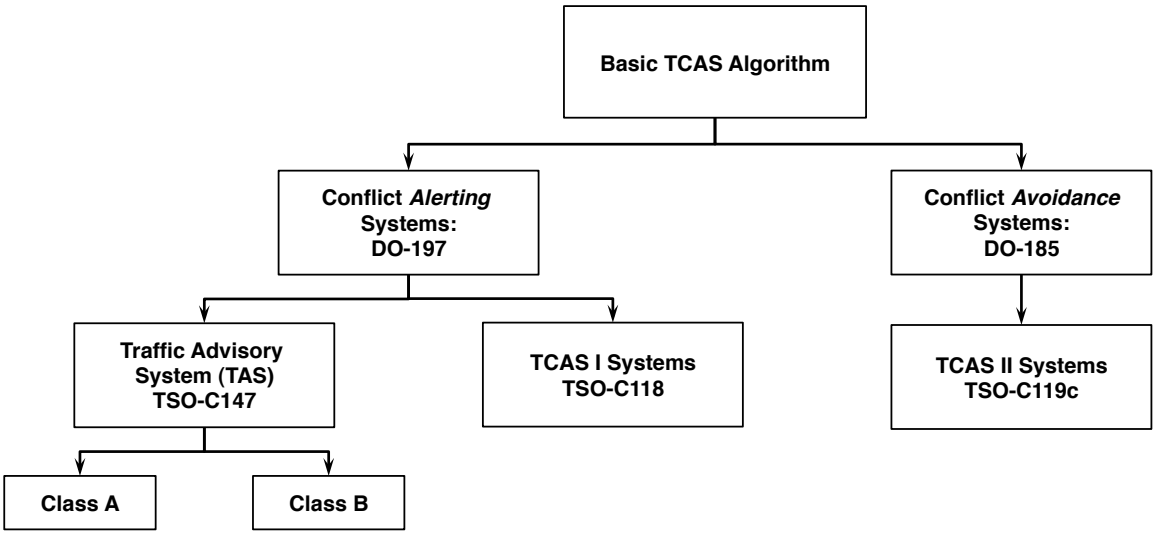


Figure 2-6: Variants of the Traffic Alert and Collision Avoidance System (TCAS)

The standards differ in which components of the algorithm are required to be implemented. Systems certified under DO-197 are intended to improve the flight crew's situation awareness by alerting them to possible future conflicts ("Conflict Alerting") in the form of Traffic Advisories (TAs). Systems certified under DO-185, in addition to alerting to conflicts,

provide executive commands to the flight crew as to how to avoid the threat aircraft ("Collision Avoidance"). Such avoidance commands are called Resolution Advisories (RAs).

In the US, a particular airborne conflict alerting or avoidance system is certified to one of three Technical Standard Orders (TSOs). All three TSOs reference the standards mentioned above and they are broken down as shown in Figure 2-6. Two TSOs exist for traffic alerting systems. Traffic Advisory Systems (TAS) systems, which are certified under TSO-C147, are intended to introduce conflict alerting to General Aviation at a lower cost than TCAS I and TCAS II systems. TSO-C147 also introduces two classes of TAS:

> ***Class A:*** *Equipment incorporating a horizontal situation display that indicates the presence and relative location of intruder aircraft, and an aural alert informing the crew of a Traffic Advisory (TA).*
>
> ***Class B.*** *Equipment incorporating an aural alert and a visual [cue] informing the crew of a TA.*

In summary, the main differences between the two TAS classes and a TCAS I system is in how alerts are presented to the flight crew visually and aurally. Table 2-1 summarizes these differences.

Table 2-1: Differences in How Alerts Are Annunciated to the Pilot for TAS and TCAS I Systems

TCAS Variant	Visual Presentation Requirement	Technical Standard Order (TSO)
TAS Class A	Traffic Display	C147
TAS Class B	"Visual Cue" for duration of alert	C147
TCAS I	Visual presentation of *Bearing* to traffic	C118
TCAS II	Traffic Display	C119c

According to Federal Aviation Regulation (FAR) §121.356, any aircraft with between 10 and 30 passenger seats must be equipped with a TCAS I system. TCAS II systems are mandated on aircraft with more than 30 seats or with a maximum takeoff weight of more than 15,000kg [48–53]. In the case of an encounter between two TCAS II–equipped aircraft, the two TCAS systems coordinate the avoidance commands they issue to the flight crew to ensure that the commands effectively resolve the situation. However, as TCAS II systems provide executive guidance to the crew, their certification costs are higher and thus the systems are generally much more expensive.

TCAS systems use an on-board sensor that actively interrogates the transponder of surrounding aircraft and performs relative state measurements of range and range rate. The sensor also measures the azimuthal reference (or bearing) between the aircraft but does so with significant error. This poor sensor performance for bearing measurements is one reason why TCAS II avoidance commands are only issued in the vertical dimension. It is also partially responsible for the failure of a mid-1990s effort to develop TCAS III, which would have provided horizontal avoidance commands [54–57]. Another limitation of the TCAS logic was identified by recent evaluations, which have shown that the TCAS II logic for generating avoidance commands is not suitable for coordination with General Aviation (GA) aircraft due to GA aircrafts' ranges of performance characteristics [58], [59].

Aside from TAS and TCAS I and II, the Traffic and Collision Alert Device (TCAD) and the Traffic Information Service (TIS) are two alerting systems commonly used in GA. TCAD is similar to TCAS I except that it uses a passive sensor instead of an active sensor to generate state measurements [52]. The passive sensor does not actively interrogate the transponders of surrounding aircraft but instead passively listens to their replies generated in response to interrogations from the ground or third-party aircraft. TCAD still uses the basic TCAS algorithm to determine when to issue alerts. Removing the need for an active surveillance sensor significantly reduces the cost of a TCAD system compared to a TAS or TCAS I system. However, without an active interrogation sensor, TCAD is dependent on external sensors such as ground based radars to interrogate the transponders of the surrounding aircraft. TIS is a data link system that uses specially equipped Mode S surface radars to uplink radar surveillance to the own-ship every 5 seconds [48].

A lower cost traffic alerting system called FLARM (FLight alARM) has been introduced in Europe, New Zealand and other parts of the world. A proprietary technology, FLARM uses an integral GPS and barometric sensor to determine position and altitude and then broadcasts that information in addition to a predicted future 3D flight path. FLARM uses a different frequency than transponders and thus only works between FLARM-equipped aircraft [60].

2.4 Introduction of ADS-B as Part of the Next Generation Air Transportation System (NextGen)

Around the same time that TCAS III development was halted, new concepts in Air Traffic Management (ATM) were being proposed that would take advantage of advancements in aircraft surveillance and navigation. Partially motivated by operations approaching capacity in the US national airspace system (NAS) and the fact that the infrastructure was based on

technology from the 1950s, the proposed new concepts were intended to improve the safety and efficiency of operations drastically by introducing Automatic Dependent Surveillance-Broadcast (ADS-B) [61]. Part of a worldwide effort to modernize the Air Traffic Control (ATC) systems, ADS-B will be the basis of the future aircraft surveillance system in the US, supplemented by the current radar system [4]. A high level overview of the US ADS-B system is provided here; Appendix A offers a more in-depth discussion of the ADS-B system architecture.

Figure 2-8 is a schematic representation of the US ADS-B system. ADS-B takes advantage of the fact that most modern aircraft have advanced navigation systems that use the global navigation satellite system (GNSS) and are often capable of determining the aircraft's position and velocity much more accurately than ground based surveillance radar. Aircraft equipped with ADS-B avionics broadcast this more accurate information and thus provide surveillance information with higher position and velocity accuracy, direct heading information as well as geometric and barometric altitude. Transmitted once per second, ADS-B has a higher update rate than radar, which updates once every 4.8 seconds in the Terminal Area and once every 12 seconds in en-route airspace. This broadcast of ADS-B messages is defined as "ADS-B Out" and is depicted by the blue arrows in Figure 2-8.

Ground stations receiving these ADS-B messages forward them via a private network to the responsible ATC facilities to be displayed on the air traffic controller's screen. Other aircraft in the vicinity can also receive ADS-B Out messages. This capability to receive ADS-B messages on-board the aircraft is defined as "ADS-B In" (depicted by the green arrows in Figure 2-8).

Figure 2-7: Cockpit Display of Traffic Information (CDTI)

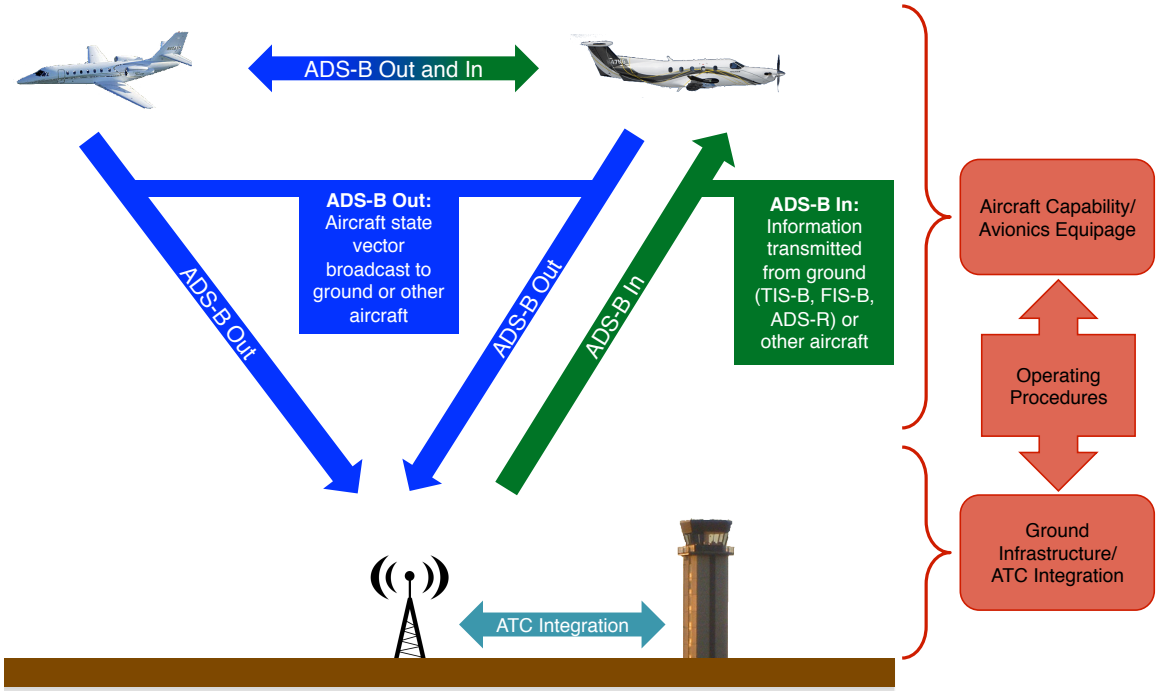


Figure 2-8: Schematic Representation of ADS-B

ADS-B In messages that originated from other aircraft can be used to display traffic in the pilot's vicinity using a cockpit display of traffic information (CDTI, Figure 2-7). In the US, ADS-B Out will be required for all aircraft operating in classes A, B, C, E above 10,000ft MSL and inside the Mode C veils of busy airports by 2020 [4].

ADS-B has a data link capability. Messages can originate from the ground stations and be used to uplink additional data directly into the cockpit of appropriately equipped aircraft. Two types of data link messages have been defined: Traffic Information Service – Broadcast (TIS-B), which provides traffic information about non-ADS-B aircraft in the vicinity of own-ship, and Flight Information Service – Broadcast (FIS-B), which provides local weather information (e.g., Doppler radar images) as well as NAS status information (NOTAMs, TFRs, etc.).

FIS-B originally was introduced to increase user benefit to GA and thus provide increased equipage incentives. However, the frequency originally proposed for ADS-B (1090MHz) had

insufficient bandwidth to support FIS-B[1]. As a result, the FAA decided to implement a dual link strategy and provide ADS-B services on two frequencies: 1090ES ADS-B, which is mostly for Air Transport and Universal Access Transceiver (UAT), and ADS-B for General Aviation [4]. Table 2-2 outlines the main differences between the two links. Note that FIS-B is only available on UAT:

Table 2-2: Differences Between 1090-ES and UAT ADS-B Link

	Mode S Extended Squitter 1090ES	**Universal Access Transceiver (UAT)**
Frequency	1090 MHz	978 MHz
Frequency shared with	TCAS, Secondary Radar, TIS-B, ADS-R	FIS-B, TIS-B, ADS-R
Intended user	Air Transport, High-End General Aviation	General Aviation
Technical standard	DO-260B, as outlined in TSO-166b	DO-282B, as outlined in TSO-154c

The decision to implement two separate links in the US introduces additional complexity to the ADS-B system: aircraft operating on one link are not able to receive ADS-B messages transmitted on the other frequency unless they are equipped with a dual-band receiver. To address this issue, Automatic Dependent Surveillance – Rebroadcast (ADS-R) was implemented. ADS-R is the capability of ADS-B ground stations to rebroadcast messages received on the UAT link to the 1090ES link and vice versa. This allows aircraft equipped with ADS-B In to receive ADS-B Out messages from aircraft on the other link with an additional one second delay. A schematic representation of the three different ADS-B traffic data sources in the US is provided in Figure 2-9.

Introducing UAT also has implications on an international level. The international ADS-B standard is the 1090ES link; any aircraft with UAT ADS-B avionics has to follow special procedures to leave the US since it does not comply with the international 1090ES ADS-B standard.

[1] 1090MHz is the interrogation reply frequency for ground based radar. Also, TCAS operates on that same frequency. There are concerns that adding ADS-B, TIS-B, and FIS-B to 1090 would overly congest it and thus reduce the efficiency of TCAS and radar.

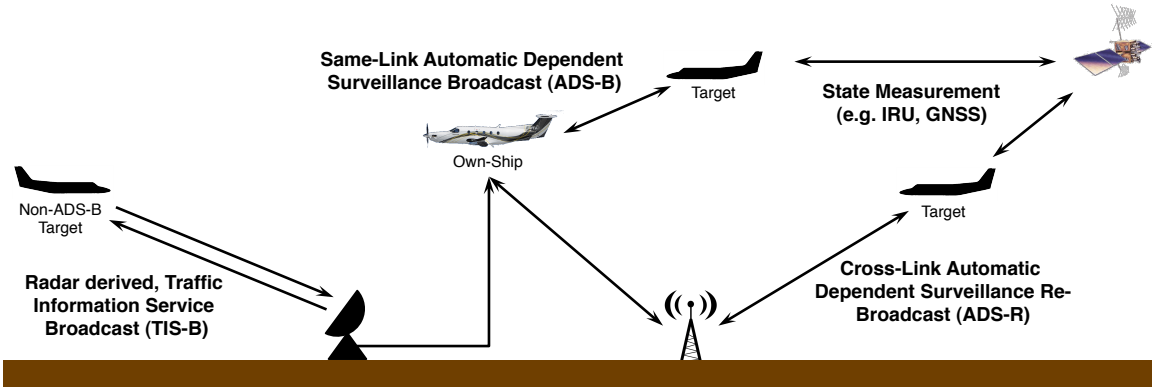


Figure 2-9: Summary Schematic of ADS-B System

2.4.1 Co-Dependency of ADS-B User Benefits and ADS-B Mandate

Much of ADS-B's benefit results from the fact that any appropriately equipped aircraft can receive ADS-B transmissions from other ADS-B equipped aircraft in the vicinity (via ADS-B In). As such, a given user's benefit from ADS-B depends on the level of equipage in other aircraft. Thus, a minimum threshold of system-wide aircraft equipage is required to justify changes in aircraft operation and ATC procedures. Ensuring equipage across all stakeholders to reach this threshold is paramount to garnering benefit from ADS-B [62]. Recognizing the need to ensure high equipage levels, aviation authorities in several countries have published mandates requiring all aircraft operating in busy airspace to equip ADS-B avionics and transmit ADS-B Out messages. This mandate takes effect in 2020 in the US and in 2017 in Europe.

Table 2-3 lists a subset of the message elements that the ADS-B mandate requires; appendix A contains a table listing all the required elements required and their performance requirements.

Table 2-3: Subset of ADS-B Message Elements Required by the Mandate and Their Minimum Performance Requirements

ADS-B Message Element	Minimum Requirement	Notes
Length and Width of Aircraft	-	Hardcoded, transmitted on ground
Latitude and Longitude	NACp 8	In reference to WGS84
Barometric Altitude	TSO-C10b certified	In 25ft or 100ft Increments
Aircraft Velocity	NACv 1	In m/s, not knots
ATC Transponder Code	-	Entered via same interface as transponder
Aircraft Call Sign	-	Either N-number or Airline Call Sign

In addition to the point estimates of position and velocity, ADS-B also contains information of how accurate each estimate is expected to be. As stated in the ADS-B standard, 95% position accuracy is defined as the "radius of a circle in the horizontal plane [...], with its center being at the true position, which describes the region assured to contain the [ADS-B transmitted] position with at least a 95% probability." [63] The radius is commonly referred to as the horizontal figure of merit (HFOM) for the position determined by GNSS receivers. Figure 2-10 shows a schematic representation of the HFOM and how it relates to the 95% position accuracy. When transmitted via ADS-B, HFOM is mapped to a set of Navigation Accuracy Category values (NACp, "p" stands for "position") as shown in Table 2-4. In the US, the ADS-B mandate requires a minimum NACp of 8, which corresponds to a HFOM of less than 93 m. In Europe, the mandate requires a minimum performance of NACp of 7, or 0.1 nautical miles.

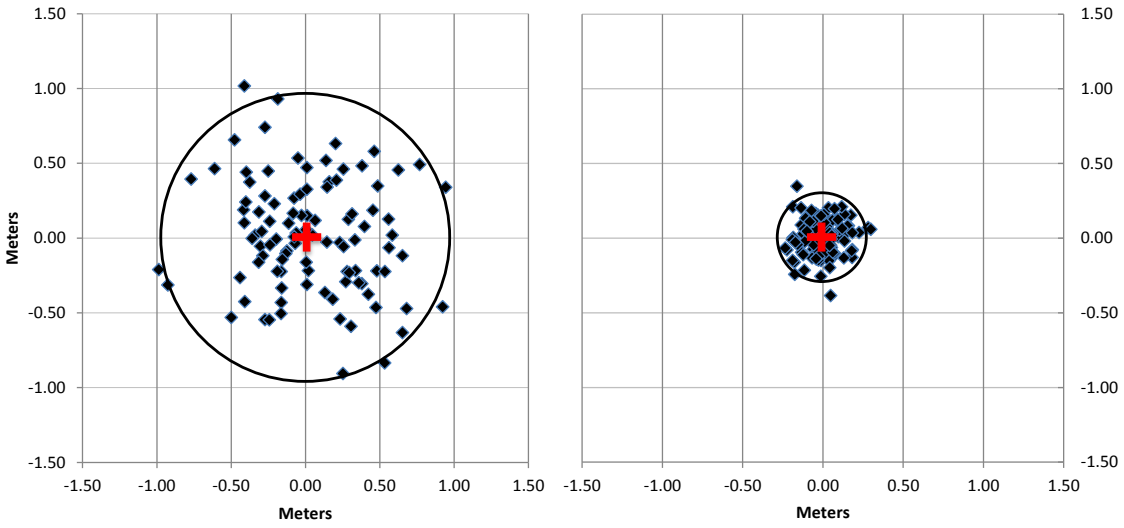


Figure 2-10: Schematic Representation of 95% Position Accuracy of 1m (left) and 0.25m (right)

Table 2-4: Mapping Between Horizontal Figure of Merit (HFOM) and ADS-B NACp Values

HFOM	NACp Value
> 10nm (18.5 km)	0
< 10nm (18.5 km)	1
< 4nm (7.4 km)	2
< 2nm (3.7 km)	3
< 1nm (1.8 km)	4

< 0.5nm (926 m)	5
< 0.3nm (556 m)	6
< 0.1nm (185 m)	7
< 0.05nm (93 m)	8
< 30 m	9
< 10 m	10
< 3 m	11

Similarly, ADS-B messages include an estimate of the accuracy of velocity as a NACv value. The accuracy value represents the range of velocities within which the true velocity lies with 95% probability, as shown in Table 2-5. The velocity accuracy required by the US and the Europe and the ADS-B mandate is a NACv of 1.

Table 2-5: Mapping Between Horizontal Figure of Merit (HFOM) and ADS-B NACp Values

Velocity Accuracy	NACv Value
>10m/s or unknown	0
±10m/s	1
±3m/s	2
±1m/s	3
±0.3m/s	4

It is important to note in regards to the methods used to encode velocity accuracy that actual GPS horizontal velocity performance is typically better than rule-compliant velocity (10 m/s) and frequently is better than 3 m/s (NACv of 2). However, many rule compliant installations will only report a NACv of 1 since few GPS receivers report the dynamic velocity accuracy metrics required for encoding a NACv of 2. In the case of a TIS-B target, the expected velocity accuracy is on the order of 19 m/s, which would be encoded as a NACv of 0 [64].

As discussed earlier, this mandate is a means to achieve the required level of ADS-B equipage by 2020. In the meantime, however, there is interest in generating incentives for airspace users to equip with ADS-B voluntarily ahead of the mandate and also in non-rule airspace [6]. One way to do this is to provide significant benefit to airspace users; the more benefit a user perceives ADS-B to provide, the more likely the user is to equip ADS-B. As mentioned, ADS-B–enabled conflict alerting has been shown to provide significant benefit and therefore its development and introduction should be accelerated. This observation is

what originally motivated the development of the Traffic Situation Awareness with Alerting Application (TSAA) [8].[2]

2.5 ADS-B and Conflict Alerting Systems

2.5.1 Early Research Related to ADS-B based Collision Alerting Systems

Free Flight was one initial concept that would take advantage of the introduction of ADS-B. Free Flight proposed a move away from structured, airway-based flight trajectories and a transfer of separation responsibility from Air Traffic Control (ATC) to the pilot when the aircraft was not in high-density environments. Using support from automation, pilots then would be able to change routes mid-flight without increasing ATC's workload. Under Free Flight, conflict detection and avoidance would play a significant role to ensure that pilots' flight path changes did not result in unsafe conditions or cause a "domino effect" of additional conflicts [65–68].

As a result of this increased focus on conflict alerting and in light of the limitations that had been discovered during the failed development of TCAS III, new approaches to conflict alerting started to emerge, some of which were summarized in section 2.2.3. However, as the steps to transition the NAS to Free Flight were being mapped out, it became apparent that Free Flight as a concept was years away from implementation due to technical and institutional challenges. The redistribution of separation responsibilities under Free Flight presents a fundamental paradigm shift that will require rigorous verification and validation before it can be implemented on a large scale in the NAS [6], [15], [69], [70]. Instead, a step-wise approach known as the Next Generation Air Transportation System (NextGen) that slowly shifts some of the responsibilities from a ground controller to the flight deck is being pursued [71], [72].

NextGen still requires conflict alerting and avoidance capability to improve beyond TCAS systems to enable some of its more advanced concepts. In order to achieve this, alerting systems must use the more accurate information available via ADS-B as validated independently via active surveillance [73]. Recent work at MIT Lincoln Laboratory related

[2] In addition to rule compliant ADS-B avionics, a parallel effort is underway to develop a standard for Low Power Surveillance Equipment (LPSE). LPSE avionics are intended for aircraft that are not required to carry ADS-B avionics under the mandate, such as sailplanes, para-gliders, ultra-lights, etc. By introducing an LPSE standard, very low-cost ADS-B units can be introduced at a significant safety benefit to the overall ADS-B system and community.

to replacing TCAS II under the Airborne Collision Avoidance System (ACAS) X program frames the decision of whether to alert as a partially observable Markov decision process. The decision to alert and the optimal collision avoidance strategy is determined using dynamic programming and based on the expected value of issuing the avoidance command [58], [74–80]. The process is applied to the design of unmanned aircraft conflict avoidance systems as well [81–83]. To reduce the number of independent validations via active surveillance, Kochenderfer et al. propose using a partially observable Markov processes to frame the decision problem of when to validate an ADS-B position report with a separate query [84]. Additional work related to replacing TCAS II at MITRE under the NextCAS program developed an approach that scales protected zones based on observed uncertainties in the given situation [85], [86].

2.5.2 TSAA ADS-B Application in the Context of NextGen

Most efforts described in this section have focused on the replacement of collision avoidance systems such as TCAS II (Figure 1-1 and Figure 2-6). However, ADS-B also presents a technological opportunity to introduce high precision conflict alerting systems to replace TAS and TCAS I systems. Combining this technological opportunity with the fact that conflict alerting provides a significant equipage incentive to GA, developing ADS-B–enabled conflict alerting has become a focus of airworthiness authorities in the US and Europe as well as industry. Known as Traffic Situation Awareness with Alerting (TSAA) ADS-B Application, this application is intended solely to improve a flight crew's situation awareness by issuing alerts on aircraft that pose a potential threat to the own-ship.

2.5.3 Development of a Certification Standard for TSAA

For aircraft to be able to equip certified TSAA systems, a Technical Standard Order (TSO) against which TSAA systems and their installation can be certified must be developed and published. The FAA as a governing body is responsible for publishing TSOs in the US. However, for the FAA to be able to publish a TSO, the TSO must first go through a process wherein the public is allowed to provide feedback on the proposed regulation. One approach for the FAA to solicit public comment is by consulting with a Federal Advisory Committee (FAC) called into existence by Congress. A FAC consists of industry experts and its main purpose is to provide advice to federal agencies on topics within their charter.

In the case of TSOs for aircraft avionics, the FAC that is frequently consulted is RTCA, formerly known as the Radio Technical Commission for Aeronautics.

For the TSAA project, the FAA is evaluating a new development approach that combines prototype development and certification standard writing in an effort to make the standard more applicable and better informed. As such, the TSAA development project serves as a

pathfinder application that will help inform how future ADS-B applications will be developed and certified. Additionally, the prototype system and exemplar algorithms developed during this project will serve as the basis for the standard published by RTCA and as a means of compliance with the standard.

Chapter 3

DEFINITION OF HIGH-LEVEL TSAA SYSTEM REQUIREMENTS

The intended function of Traffic Situation Awareness with Alerting Application (TSAA) is to provide timely alerts on qualified airborne traffic in the vicinity of the own-ship in order to increase the flight crew's traffic situation awareness[3]. Thus, TSAA reduces the risk of a near mid-air or mid-air collision by enhancing the flight crew's situation awareness and ability to see-and-avoid. This chapter translates this intended function into a set of high-level system requirements. The high-level system requirements are grouped into the following categories:

- **Stakeholder requirements**, which are informed by the expectations of the involved stakeholders
- **Functional Requirements**, which are informed by the operations that the system will encounter once commissioned
- **Architectural Requirements**, which define the system architecture
- **Performance Requirements**, which define acceptable system performance

The following sections will address each of these categories individually.

3.1 Identification of Stakeholder Requirements: General Aviation Users and Airworthiness Authorities

The context that motivated the development of TSAA was the US Federal Aviation Administration's (FAA) desire that ADS-B provide early benefit to airspace users. In doing so, the FAA intends to generate incentives for users to equip with ADS-B avionics ahead of

[3] This definition is provided in the ADS-B Integrated Working Plan (AIWP). Written by a committee of industry and government representatives, the plan puts forth a roadmap of how and in what order the various ADS-B applications and functionalities will be implemented. MIT participated in the generation of the plan [6].

the 2020 ADS-B Out mandate and in airspace where equipage is not mandated. However, it is important to remember that TSAA as a system will not be specific to the US, but rather will be used and standardized worldwide. At a high level, therefore, the two groups of stakeholders that must be evaluated in depth are the airspace users and airworthiness authorities worldwide.

3.1.1 General Aviation as a Group of Stakeholders

Two of the major airspace users that operate aircraft in the National Airspace System (NAS) are Commercial Aviation (FAR Part 121 operators) and General Aviation (GA; e.g., Part 91 or 135). In the US, GA makes up over 96% of all active aircraft. Figure 3-1 shows the Bureau of Transportation Statistics (BTS) record of all active aircraft from 1960 to 2011. In this plot, the GA aircraft consist of aircraft operated under Part 135 (on-demand operations) and Part 91 regulations.

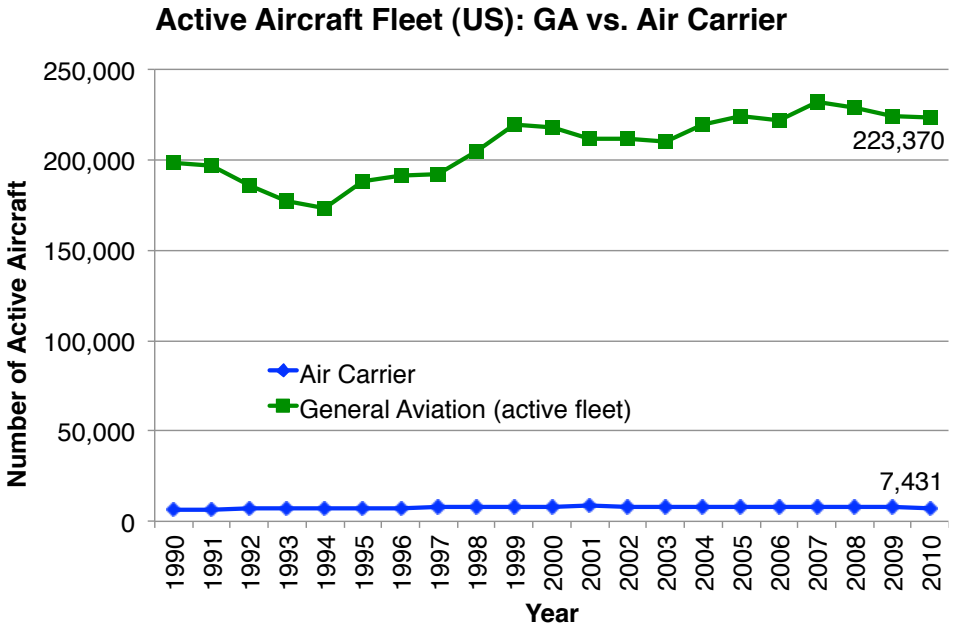


Figure 3-1: Comparison of General Aviation to Air Carrier Active Fleet. General Aviation Includes Air Taxi (BTS)

Although GA aircraft vastly outnumber air carrier aircraft, yearly GA aircraft use is much lower, as shown in Figure 3-2. The average hours air carrier aircraft have flown per year have increased over recent years to 2,387 hours while GA aircraft use has remained mostly steady, with an average of 97 hours flown in 2010. Thus, the average yearly use of an air carrier aircraft is 25 times higher than that of a GA aircraft.

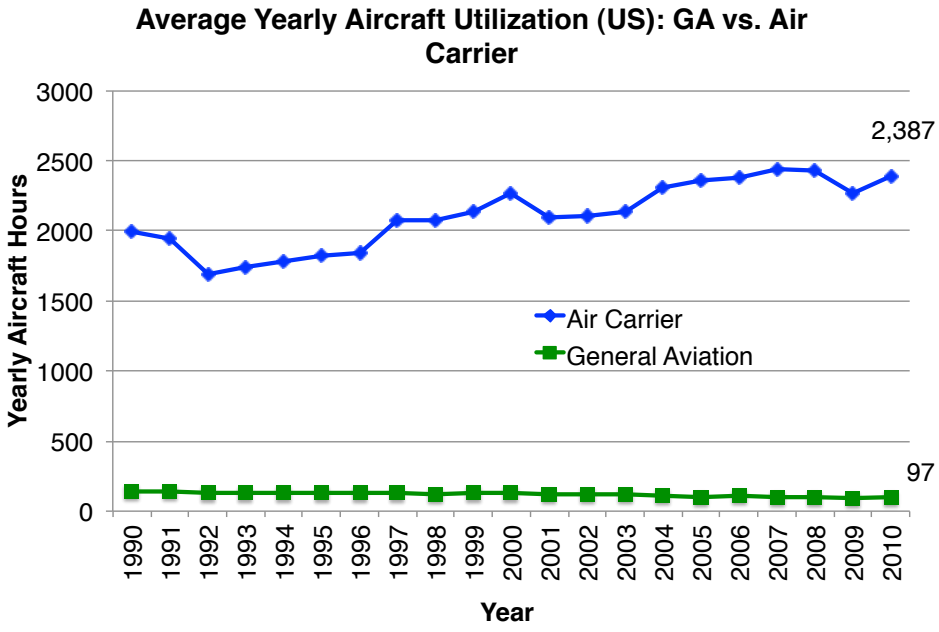


Figure 3-2: Average Yearly Hours Flown by General Aviation Aircraft compared to Air Carrier Aircraft (BTS)

User benefit can be differentiated into multiple categories, including increased operational efficiency and improved safety. Users who use aircraft more frequently benefit from efficiency gains in their operations, which can offset the cost of the avionics significantly. As such, commercial aviation is more likely to equip with ADS-B voluntarily and early.

For GA users, the business case is less likely to resolve solely based on efficiency gains due to lower aircraft use. Furthermore, GA tends to be more sensitive to cost because the aircraft owners often pay expenses incurred from new equipment out-of-pocket. However, the business case for GA can, be improved by introducing safety benefits such as a lower risk of mid-air collisions. Unlike commercial operators, most GA aircraft do not have a conflict alerting system – thus, providing a low-cost, ADS-B–enabled system capable of alerting on all types of operations, including those of GA users, generates an incentive for them to equip ADS-B avionics [8], [9].

IMPROVING THE COST/BENEFIT CASE FOR GENERAL AVIATION

Much of the cost of new avionics equipment is linked to the cost of certification. Systems that provide avoidance guidance to the flight crew carry a higher certification burden than systems that only issue alerts and do not direct the crew. Thus, from a cost perspective, it is more attractive to make TSAA an alerting system rather than an avoidance system.

From the benefit perspective, it is important to note that GA in particular comprises a very diverse range of operations and aircraft. As such, GA is more of a group of stakeholders than one single stakeholder. Figure 3-3 shows the percentage of General Aviation aircraft in blue by primary use and age. Red represents the percentage of GA hours flown [3]:

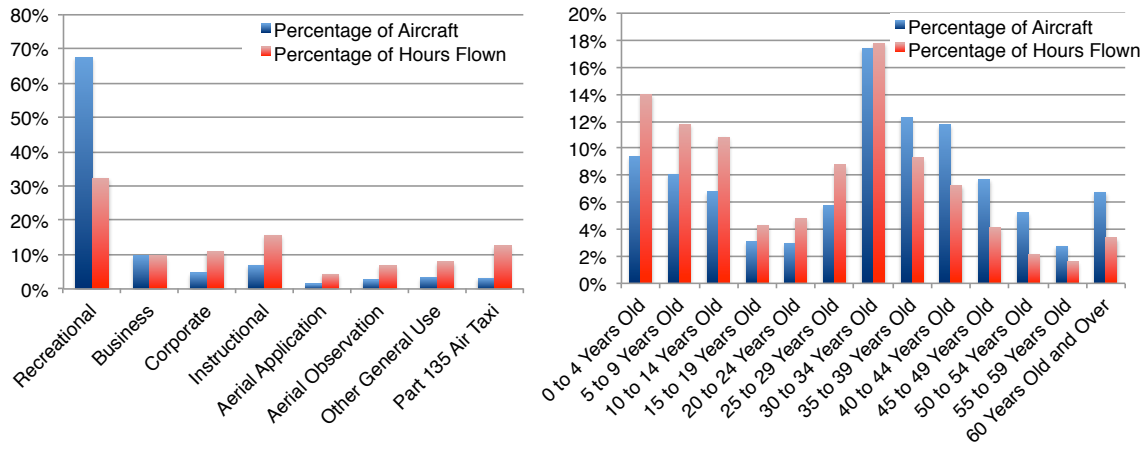


Figure 3-3: Percentage of General Aviation Aircraft By Primary Use and Age

As Figure 3-3 shows, GA aircraft are used for a large variety of purposes. Recreational, instructional, or aerial observation operations tend to be highly dynamic and may stay in the same area for the entire duration of the flight. In contrast, business and corporate operations tend to maintain constant heading and altitude over long distances. As discussed in section 2.2.2, as operations become more dynamic, future state uncertainty increases. Therefore, for TSAA to maximize benefit to GA, it must be robust to high levels of future state uncertainty, which will allow the system to perform reliable alerting in highly dynamic operations.

Figure 3-3 also demonstrates that recreational/personal use is most common purpose for which GA aircraft are used, and that most GA aircraft are over 20 years old. This underscores the requirement to keep the cost for avionics low as many owners must pay for these avionics out-of-pocket for aircraft that have reduced value due to depreciation [3].

A second observation with respect to increasing benefit is that as operators begin to equip aircraft with ADS-B avionics, there will be a period of mixed equipage. Additionally, after 2020, not all aircraft will be equipped with ADS-B, as it will not be required in all airspace. The percentage of unequipped aircraft is likely to be higher among GA for the reasons discussed above. If such unequipped aircraft are in view of a ground based radar, they will

be visible to other aircraft via TIS-B or alerting systems with active interrogation sensors[4] (see section 2.4). If the unequipped aircraft is not in view of a ground based radar, no target information will be available for that aircraft via ADS-B. Additionally, if the own-ship is operating outside of the surveillance volume, it will also not be able to receive TIS-B messages, even if the targets are in sight of a ground based radar. According to a survey by Lester and Hansman, over two thirds of pilots spend at least 10% of their operating time outside of radar coverage [9].

This leads to 2 conclusions. First, it demonstrates the importance of a strong cost-benefit case for GA in order to increase GA equipage. A strong cost-benefit case helps reduce the total number of aircraft that are not equipped with ADS-B and would thus be invisible outside of radar coverage. Second, it is likely that GA operators will encounter TIS-B targets more frequently while within radar coverage. Thus, an alerting system capable of alerting on TIS-B targets would provide additional benefit to GA users.

Based on this discussion, the following high-level GA stakeholder requirements can be derived:

1. The equipage costs shall be minimized
2. TSAA shall be an alerting system and not an avoidance system
3. TSAA shall be able to alert during highly dynamic operations
4. TSAA shall be able to alert on all ADS-B targets, including TIS-B (radar derived) targets

3.1.2 AIRWORTHINESS AUTHORITIES AND STANDARDS-SETTING BODIES

A certification standard must be developed before TSAA can be introduced and certified. In the US, one of the standard-setting bodies for aircraft avionics is RTCA. The corresponding body in Europe is the European Organization for Civil Aviation Equipment (EUROCAE). Commonly, the two organizations collaborate during the development of new standards. Once the standard is written, the civil airworthiness authorities in a particular country publish regulations that reference that standard and require aircraft to comply with it.[5] As a

[4] To be visible to secondary surveillance radar and thus via TIS-B, the aircraft must be equipped with a transponder. Currently, 96% of the GA fleet is equipped with a transponder [3]. There are plans to phase out TIS-B service after 2020 since non-ADS-B-equipped aircraft are will likely be observed infrequently in radar-airspace, removing the need for TIS-B.

[5] Note that a standard is not necessarily required for this process – for example, in the US the aircraft certification branch of the FAA can publish a stand-alone technical standard

result, the airworthiness authorities and standards setting bodies are additional stakeholders in the development of TSAA.

The FAA's goal of incentivizing early equipage of ADS-B by introducing TSAA results in three additional requirements. First, in order to introduce TSAA to the NAS as a new functionality, a certification standard must be written against which TSAA can be certified. Therefore, the exemplar TSAA system must be able to serve as a basis for developing that certification standard. Second, the time to introducing TSAA should be kept to a minimum. The earlier it is introduced, the larger the aggregate benefit and the more time remains for users to equip voluntarily before the mandate takes effect. Third, in order to generate an incentive to equip with the rule-compliant ADS-B Out avionics, TSAA must be designed to operate with ADS-B targets primarily.[6] In other words, given the earlier requirement that TSAA shall be capable of operating with TIS-B targets, that requirement should not be driving the design of the system.

In summary, the following additional high-level stakeholder requirements can be derived:

5. The exemplar TSAA system shall be able to serve as the basis for the certification standard
6. The time to the introduction of TSAA shall be kept to a minimum
7. TSAA shall be primarily designed to operate primarily with ADS-B and ADS-R targets

3.2 Identification of Functional Requirements for TSAA: Analysis of 10 years of Mid-Air Collision Data

In order to identify where the risk for a mid-air collision (MAC)—and thus the benefit potential for TSAA—is highest, an analysis on where MACs most frequently occur was conducted.

This analysis focused on documented mid-air and near mid-air collisions that occurred between 2000 and 2010. Based on the results, a set of representative scenarios was generated for the purpose of defining the TSAA functional requirements.

---

order (TSO) that does not reference a standard. Frequently, a TSO references a standard but levies requirements in addition to those in the standard to achieve compliance.

[6] By policy, the FAA does not allow the installation of a certified ADS-B In system without also requiring that a certified ADS-B Out system be installed on the same aircraft.

3.2.1 Analysis Of NTSB Accident Reports Of Mid-Air Collisions

First, mid-air collision reports in the accident database of the National Transportation Safety Board (NTSB) reports from January 2000 to June 2010 were analyzed. Reports of accidents outside the US as well as balloon accidents that occurred during that time period were excluded. This resulted in 112 accident reports. The reports did not contain any mid-air collisions involving an aircraft operating under instrument flight rules (IFR) or Part 121 regulations.

The narrative of each of the 112 reports was reviewed and the encounter geometry was reconstructed for each mid-air collision. The method of describing aircraft heading differed between reports (Table 3-1): some reports gave exact headings, others used cardinal directions (North, Southwest, etc.) and others only described the location of the aircraft relative to each other. Some reports did not have any radar data or eyewitnesses available and thus did not have flight track information at all. To allow for the comparison of the horizontal encounter geometries, the accidents were grouped into bins of 45° based on flight track intersection angle. The 5 groups were centered on the 5 cardinal directions of one half of a compass rose (see Figure 3-5). In addition to geometry reconstruction, external factors that contributed to the collision were identified (such as the absence or malfunction of equipment).

Table 3-1: Format of Heading Information in NTSB Mid-Air Collision Reports

Description of Heading	Percentage
Cardinal Directions	19%
Exact Radar Data	11%
Implied from description in report	63%
No heading information available	7%

The description of vertical motion of the aircraft was much less consistent. Many reports never mentioned vertical movement, while others simply stated that the aircraft was climbing or descending. In many cases, however, it was possible to extract at least the relative vertical motion of the two aircraft based on the narratives.

3.2.2 Analysis Of ASRS And NMACS Database Near Mid-Air Collision Reports

In an effort to widen the scope and validate the findings of the NTSB report analysis, the analysis was expanded to include Aviation Safety Information Analysis And Sharing (ASIAS) reports of near mid-air collisions. The ASIAS Aviation Safety Reporting System (ASRS) and ASIAS Near Mid-Air Collision System (NMACS) databases were searched for every event

classified as a near mid-air collision (NMAC) during the same time period used for the NTSB report analysis. The ASRS database yielded 2,059 reports and the NMACs database yielded 1,527 reports. The reports in the ASRS database contain both a narrative of the event and set of fields filled in by the report's creator. The reports in the NMACS database contain a similar set of data fields but do not include a publicly available narrative. The data fields were analyzed for the frequency that a given value appeared. For example, the reported flight phases of the own-ship were plotted versus the reported flight phases of the target aircraft.

One interaction that was not observed in the NTSB database was encounters between commercial and GA aircraft. However, this interaction was present in the ASRS and the NMACS databases; thus, a secondary analysis of GA/Part 121 encounters in those databases was conducted. Aircraft operating under Parts 91, 135, 137 and 141 were all considered GA.

Since these databases are voluntary reporting systems, interpreting the results requires caution. Filing an ASRS report gives the reporter certain protections against possible charges and as such creates a reporting bias toward events in which the pilot violated a regulation[7]. Also, because of their subjectivity, the reports "…represent what the reporter believes he/she saw or experienced." Lastly, a cross analysis showed that IFR report rates are higher than the percentage of IFR hours flown, which indicates some over reporting or higher sensitivity in the IFR population.

3.2.3 RESULTS FROM NTSB REPORT ANALYSIS

LOCATION ANALYSIS OF NTSB ACCIDENT REPORTS

All accidents reported in the NTSB database were separated into three categories based on their proximity to the airport (Figure 3-4). The category defined as "Pattern" only includes accidents involving aircraft that were flying the airport pattern with intention to land or that had recently departed the airport. The category defined as "Airport Vicinity" includes accidents that occurred outside the pattern but while the accident aircraft were still engaging an airport. As Figure 3-4 shows, the area surrounding an airport is where most (59%) mid-air collisions occurred. As a single category, the airport pattern was the location with the most accidents (45%). This implies that the location of highest benefit for TSAA is the airport environment, and specifically in the airport pattern.

[7] The ASRS database website notes: "The existence in the ASRS database of records concerning a specific topic cannot, therefore, be used to infer the prevalence of that problem within the National Airspace System." [106]

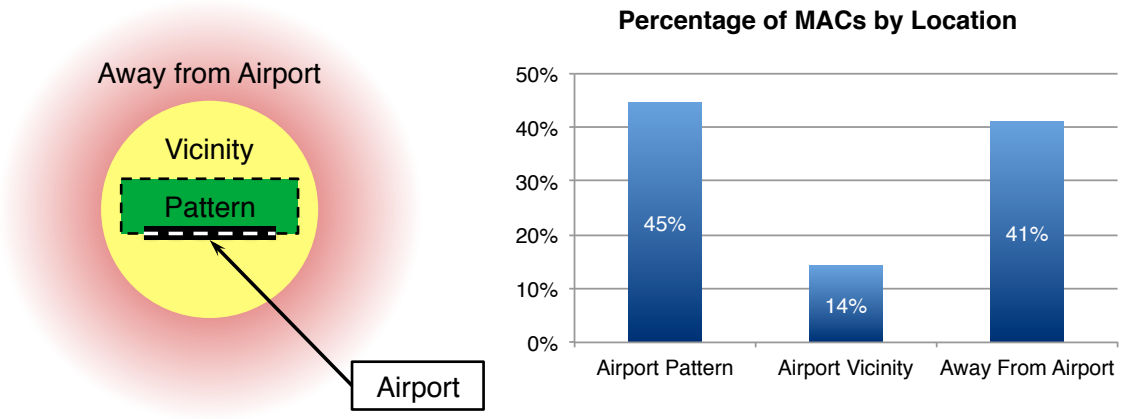


Figure 3-4: Percentage of NTSB Mid-Air Collisions by Location

GEOMETRY ANALYSIS OF NTSB ACCIDENT REPORTS

Figure 3-5 summarizes the intersect angle between the tracks of the two aircraft for all analyzed accident reports. The own-ship is the aircraft in the center and the target aircraft for a given mid-air collision is one of the aircraft along the perimeter of the compass rose. The colors and percentages indicate the frequency at which a given intersect angle was reported. As can be seen, over half (54%) of mid-air collisions occur between aircraft flying in the same direction.

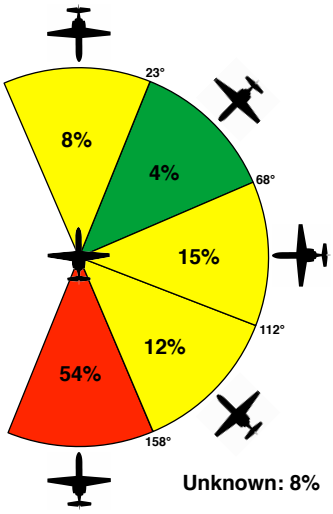


Figure 3-5: Track Intersect Angle Summarized for All NTSB Mid-Air Collision Reports

To gain a better understanding of the characteristics of these encounters based on their location, each of the three environments identified in Figure 3-4 was analyzed individually.

Detailed Analysis Of Mid-Air Collisions Reported In The Airport Pattern

Out of the 112 reported cases, 50 occurred in the airport pattern. This section analyzes those 50 accidents in more detail. As Figure 3-6 shows, over 80% of the mid-air collisions in the airport pattern took place on final, short final, or on the runway. As a result, the track intersection angle observed most often is that of two aircraft going in the same direction. The narratives of these reports paint a similar picture for most of these accidents: two aircraft on approach to the same runway settling into each other as they get closer to the runway. This type of encounter characteristically occurs at a small relative velocity, which often results in the two aircraft only "bumping" each other. As a result, 31 of the 50 accidents in the airport pattern were non-fatal.

Out of the 50 accidents, 9 (18%) involved at least one aircraft that was not equipped with a radio for voice communication. According to the 2010 FAA Avionics Survey, only 2% of the GA fleet did not have a radio installed. Six accidents (12%) involved at least one agricultural aircraft. According to the FAA Avionics Survey, agricultural aircraft fly 5% of the total hours flown by GA. As such, aircraft without radios as well as agricultural aircraft were more strongly represented than what would have been expected.

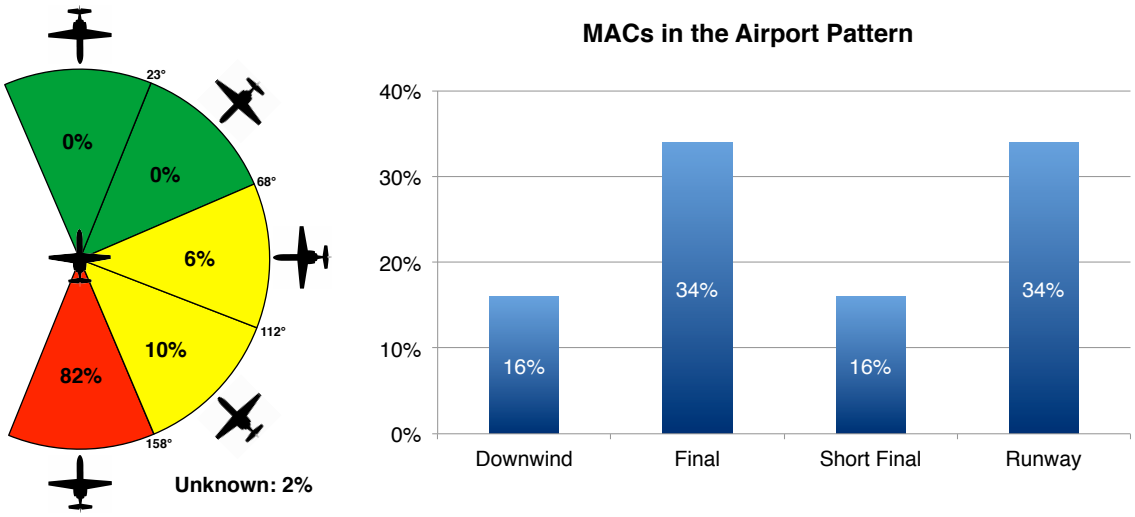


Figure 3-6: Location Distribution and Geometry of All NTSB Mid-Air Collisions in the Airport Pattern

Detailed Analysis Of Mid-Air Collisions Reported In The Airport Vicinity

A total of 16 accidents occurred in the airport vicinity. Nine of those were between aircraft that had identical flight phases, i.e., both aircraft were departing from or both arriving at the airport. Three accidents occurred inside the bounds of the airport pattern while the aircraft were not actually flying the pattern. Specifically, one collision occurred during a race, one during parachute operations, and one during practice for an airshow above the airport. The remaining 4 accidents involved one aircraft that was arriving to or departing from an

airport while the other aircraft was cruising past or performing maneuvers around that same airport. Figure 3-7 shows the geometry distribution for the accidents reported in the airport vicinity.

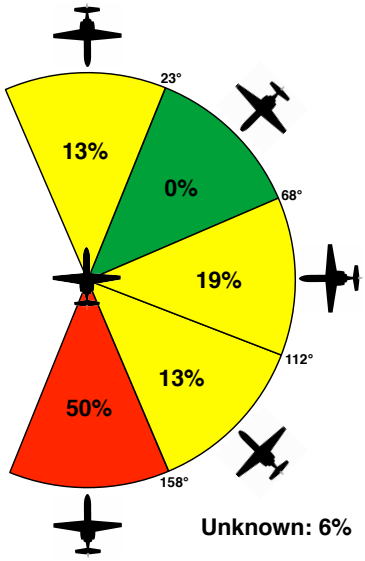


Figure 3-7: Geometry Distribution for Encounters in the Vicinity of the Airport

DETAILED ANALYSIS OF MID-AIR COLLISIONS REPORTED AWAY FROM THE AIRPORT

A total of 46 accidents occurred away from the airport. These accidents included aircraft that were in cruise as well as aircraft engaging in flight training, surveying, firefighting, EMS transport, aerial application (e.g., crop dusting), or news reporting (all of which referred to as "Maneuvering" in Figure 3-8). As Figure 3-8 shows, out of the 46 accidents, 24 (52%) occurred between two aircraft that were both in straight and level cruise. Nine (20%) were between aircraft that were deliberately engaging in close flight such as pilots practicing formation flight or friends going to a similar destination. Those accidents are labeled as "Formation Flight" in Figure 3-8 and Figure 3-9. The remaining 13 accidents (28%) involved at least one aircraft conducting maneuvers such as surveying, firefighting or flight instruction.

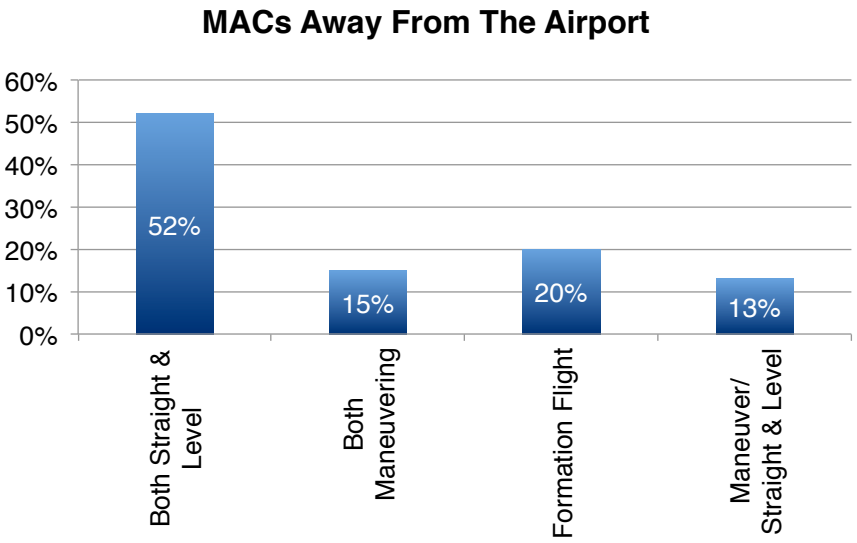


Figure 3-8: Flight Phases of Mid-Air Collisions Away From the Airport

The intersect angle observed most often is that of two aircraft with perpendicular tracks (29%), as shown in Figure 3-9. This may be due to blind spots resulting from wings and/or window frames that extend out from the side of the aircraft. A recurring theme in the narratives (6 cases) was witnesses or survivors mentioning sun glare as a contributing factor.

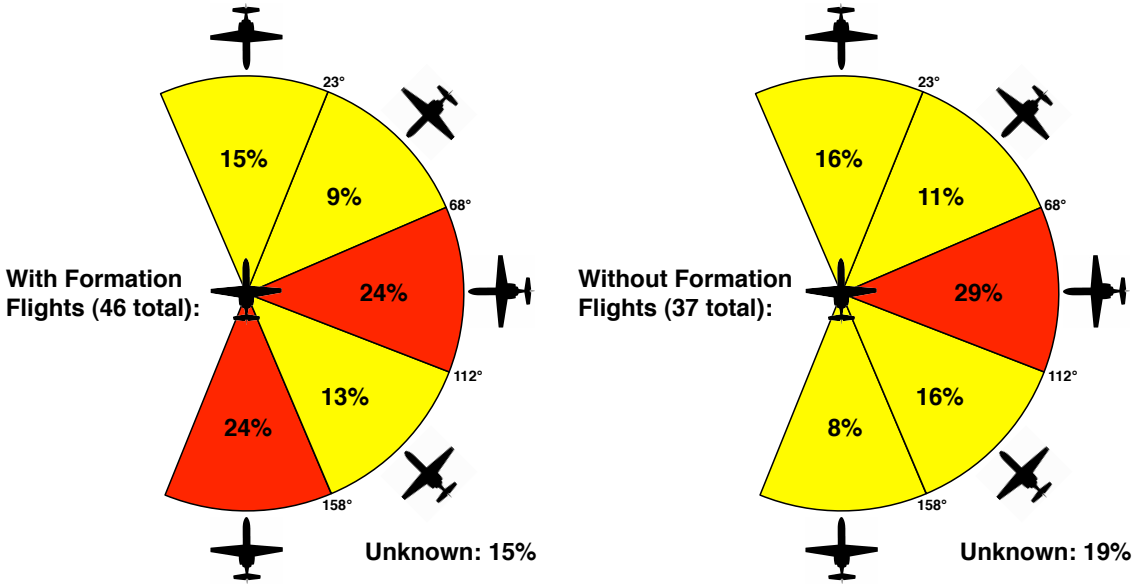


Figure 3-9: Track Intersect Angle for Mid-Air Collisions Away From the Airport With and Without Formation Flights

3.2.4 RESULTS FROM THE ASRS AND NMACS DATABASE ANALYSIS

The ASRS and NMACS databases first were evaluated based on the flight phases of the reporting and target aircraft. Reports that included a field left as "unknown" are not shown. Figure 3-10 and Figure 3-11 show the near mid-air collision reports for both databases with flight phases on the X and Y-axes. The Z-axis is the percentage of a given interaction. The flight phases on both axes are aligned such that the diagonal represents the encounters between two aircraft on the same flight phase. In the ASRS as well as the NMACS data, the flight phase interactions most often observed are those of two aircraft on "Initial Approach" (24% and 14%, respectively). Note that "Approach" is one category in the NMACS data, but is split into three sub-categories in the ASRS reports. The second most common interaction was between two aircraft in "Cruise" (11% and 13%, respectively). A review of the ASRS narratives showed that reports with flight phases categorized as "Initial Approach" were most often in the pattern.

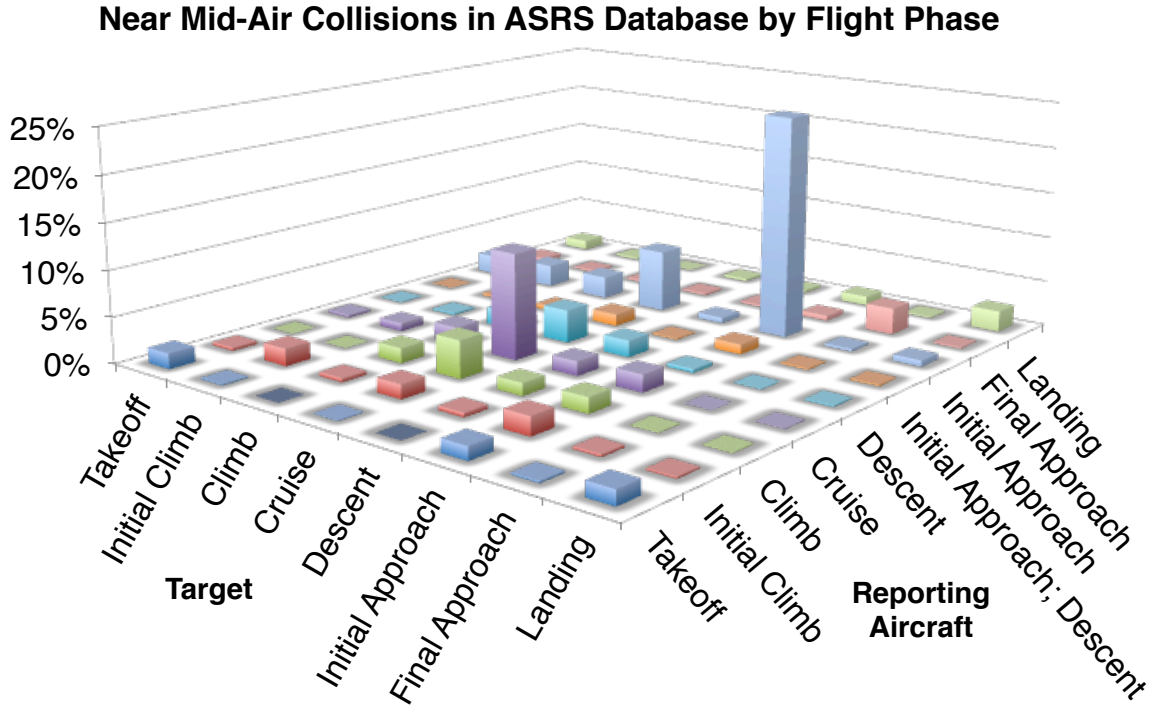


Figure 3-10: Near Mid-Air Collisions Reported in the ASRS Database by Respective Flight Phase. Encounters Along the Diagonal Are Between Aircraft in the Same Flight Phase.

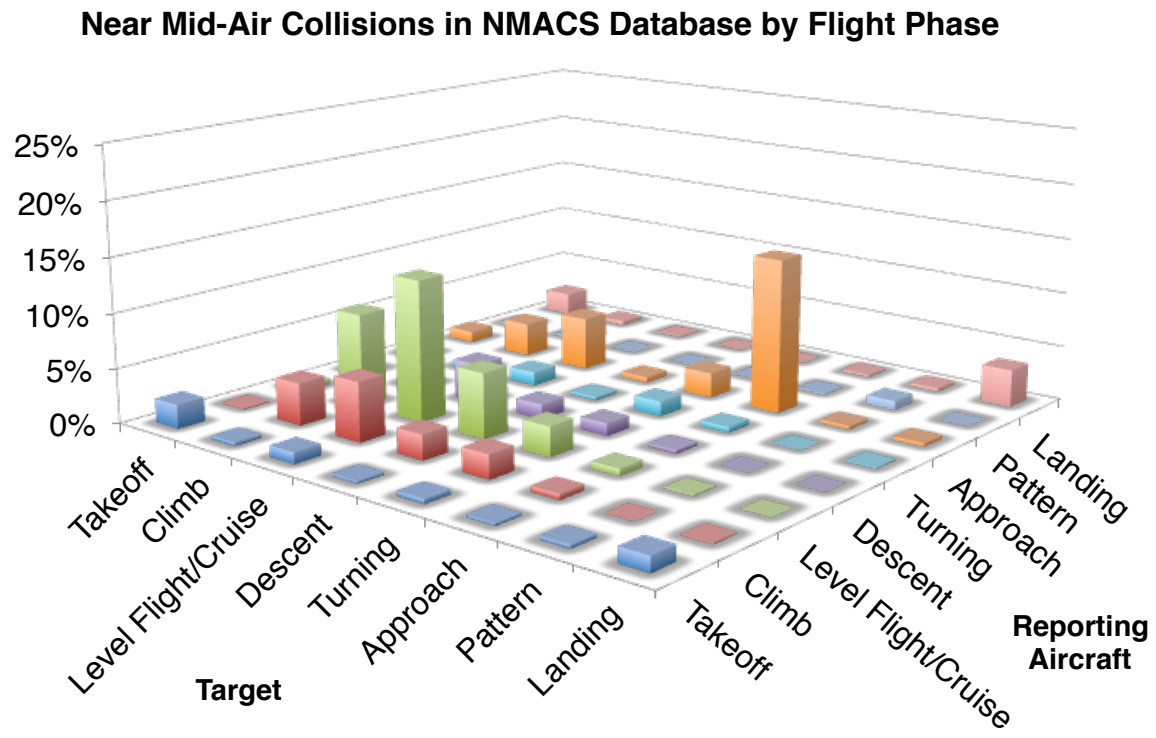


Figure 3-11: Near Mid-Air Collisions Reported in the NMACs Database by Respective Flight Phase. Encounters Along the Diagonal Are Between Aircraft in the Same Flight Phase.

Both figures underscore the observation made in the NTSB reports that the airport environment is the location where most encounters are reported. Table 3-2 shows the percentages of encounters reported in the airport environment in the ASRS and NMACS databases.

Table 3-2: Near Mid-Air Collisions Reported in the Airport Environment

Database	Percentage
ASRS	64%
NMAC	47%
NTSB	59%

Table 3-3 shows the percentages of encounters according to under which Federal Aviation Regulation (FAR) the aircraft were operating. Both databases indicate that encounters between GA aircraft are most common, which is consistent with the NTSB mid-air collision

data. However, unlike the NTSB data, interactions between GA and Part 121 aircraft were also observed in the near-miss data.

Table 3-3: NMAC Encounters by FAR, Ranked by Percentage

ASRS Database	
Interaction	**Percentage**
GA/GA	44%
GA/Part 121	14%
Part 121/Part 121	5%
At least one aircraft unknown	36%

NMACS Database	
Interaction	**Percentage**
GA/GA	28%
GA/Part 121	14%
GA/Military	8%
Part 121/Part 121	3%
At least one aircraft unknown	47%

The flight phases of the GA/Part 121 encounters are shown in Figure 3-12. The largest interaction observed in the ASRS database was between a Part 121 aircraft on "Initial Approach" and a GA aircraft on "Cruise" (20%). The data indicates that encounters are most likely when the GA aircraft is in cruise and the Part 121 aircraft is in any other flight phase, specifically climbing or descending. This is likely due to Part 121 aircraft often transitioning through altitude layers where GA aircraft would be cruising. Also shown in Figure 3-12 is the altitude distribution where the GA/Part 121 encounters took place. Again, encounters were reported most frequently at altitudes where GA most commonly operates.

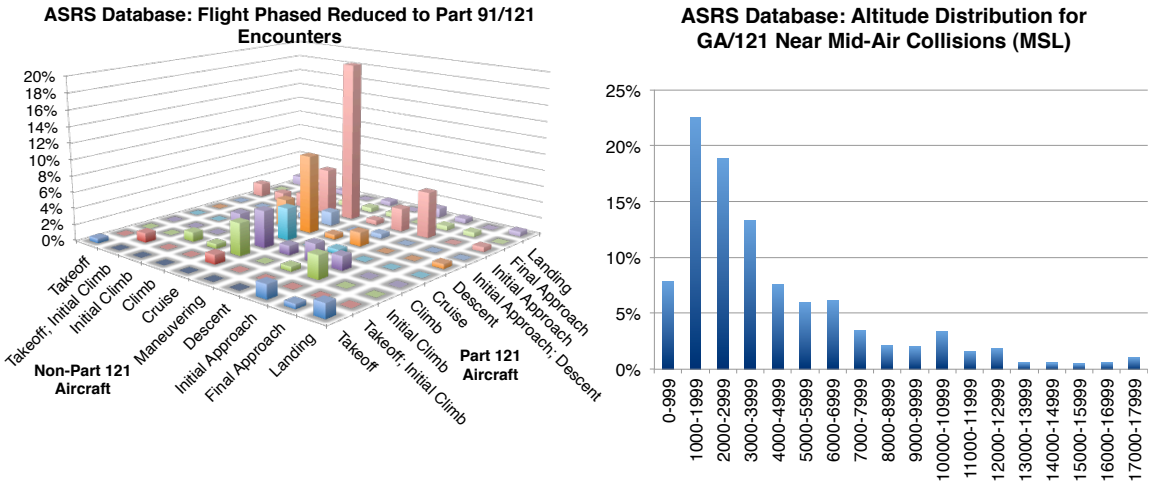


Figure 3-12: Flight Phase and Altitude Distribution of GA/Part 121 Encounters in the ASRS Database

Figure 3-13 shows the same GA/Part 121 analysis using NMACS data. Here, the largest interaction was between two aircraft on "Approach" to an airport (12.5%). The encounter between cruising/transitioning aircraft observed in the ASRS data is not as pronounced but can still be observed. Most interestingly, the altitude distribution of the NMACS reports shows two distinct peaks: One at low altitude, as observed in the ASRS database, and the secondary peak around 10,000ft MSL. Upon reviewing the narratives, the low level peak is mostly from VFR traffic while the mid-altitude peak is from cruising IFR traffic as well as sailplanes. Additionally, the second peak may be a result of increased aircraft velocities due to the airspeed restriction of 250kts below 10,000ft.

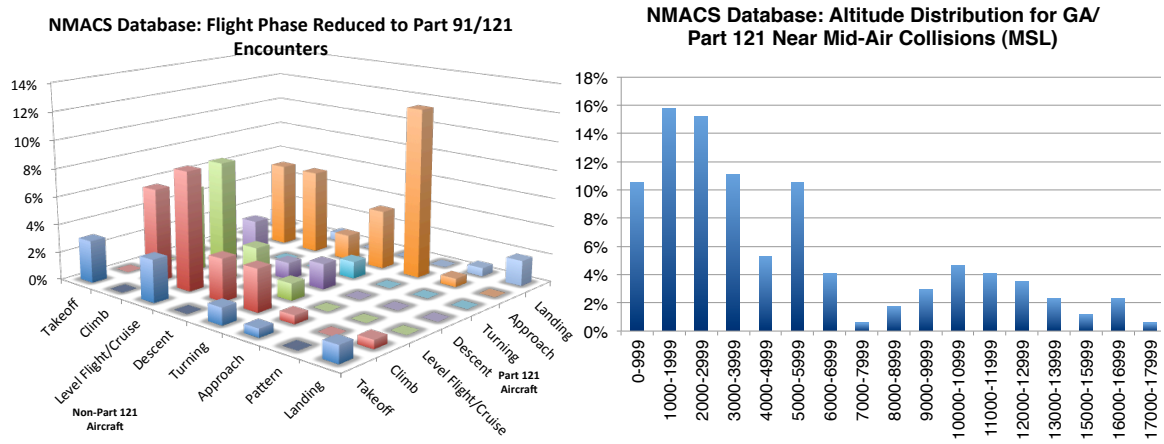


Figure 3-13: Flight Phase And Altitude Distribution of GA/Part 121 Encounters in the NMACS Database

3.2.5 DERIVATION OF FUNCTIONAL REQUIREMENTS

In summary, the airport environment is where most mid-air collisions occurred (59%) and where most near mid-air collisions were reported (ASRS, 67%). All the mid-air collisions occurred between GA aircraft, which further underscores the requirement that TSAA operates reliably during General Aviation operations.

Encounters between Part 121 and GA aircraft were most often reported to occur between GA aircraft cruising at a constant altitude and Part 121 aircraft transitioning through that same altitude. These interactions are observed most often in two distinct altitude layers: low altitude (ground to 4,000 feet MSL) and mid-level (9,000 feet to 13,000 feet MSL), as shown in Figure 3-13.

At a high level, this results in the following requirements for TSAA: it must be capable of alerting both in the airport environment as well as on transitioning commercial aircraft. In order to define the functional requirements more precisely, a set of 14 scenario categories

capturing all the mid-air collisions observed above was generated. Each encounter category includes a narrative, a percentage of how often such an encounter was observed in the databases, and a range of values for Track Intersect Angle (IA), Relative Vertical Velocity (RVV), and Relative Horizontal Velocity (RHV). Figure 3-14 represent how those three variables are defined.

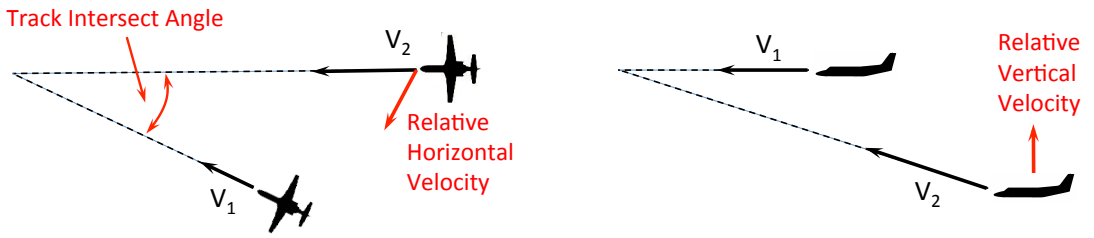


Figure 3-14: Definitions of Track Intersect Angle (IA), Relative Horizontal Velocity (RHV) and Relative Vertical Velocity (RVV)

The encounter categories are separated into encounters that occur in the vicinity of an airport (A), which include A1-A8, and those that occur in the en-route environment (E), which include E1-E6).

The functional requirement of TSAA is that it be capable of reliably alerting on all 14 scenarios categories:

1. TSAA shall reliably alert in all 14 scenario categories

SCENARIO CATEGORIES REPRESENTING OBSERVE MID-AIR AND NEAR MID-AIR COLLISIONS

The 14 categories with their narratives, parameter ranges and schematics are shown below.

A1	Two aircraft are converging on the same leg of an airport pattern. Their tracks and altitudes are similar and as a result their relative velocity is small. An example is two aircraft converging while on Final to the same runway. (Basis: NTSB, 30.5%)	RVV: < 200 fpm RHV: < 20kts IA: < 10°
A2	Two aircraft with *different* flight phases are converging on the same leg of an airport pattern. Their tracks are similar but their altitudes and vertical velocities are not. As a result, their relative velocity is larger than in scenario A1, consisting mostly of vertical velocity. An example is one aircraft climbing on downwind, having just departed, while another one is arriving via downwind. (Basis: NTSB, 6.3%)	RVV: 200 - 3000 fpm RHV: < 20kts IA: < 10°

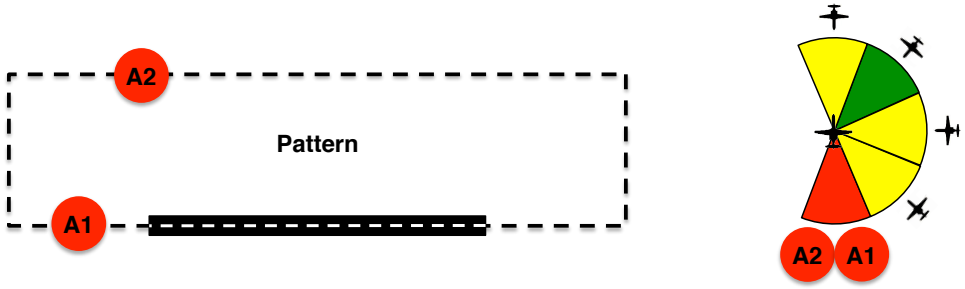


Figure 3-15: Narrative, Location and Geometry of Encounter Category A1 and A2

A3	One aircraft is established in the pattern and a second aircraft is joining via a standard procedure (such as 45° entry). (Basis: NTSB, 6.3%)	RVV: 0 - 3000fpm RHV: 0 - 200kts IA: < 60°
A4	One aircraft is established in the pattern and a second aircraft is joining via a non-standard procedure (such as a direct final or left/right patterns). (Basis: NTSB, 10.5%)	RVV: 0 - 3000fpm RHV: 0 - 200kts IA: 60° - 180°

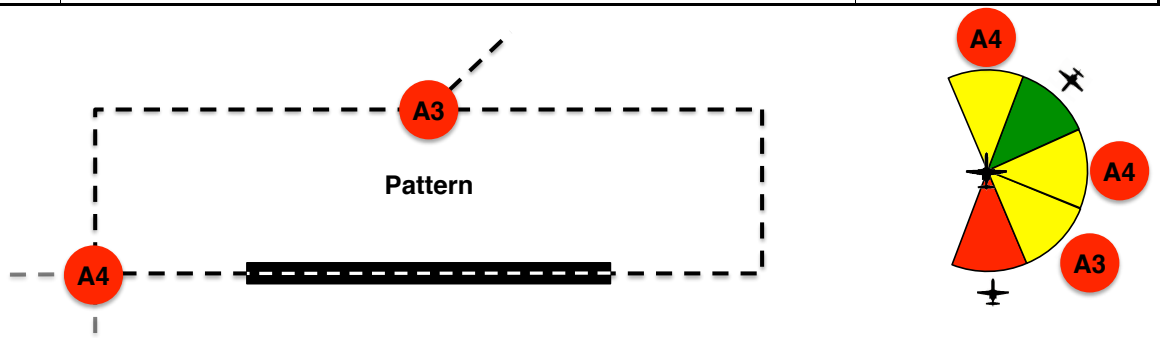


Figure 3-16: Narrative, Location and Geometry of Encounter Category A3 and A4

A5	A VFR GA aircraft is cruising above pattern altitude (e.g., 2,500 ft). A departing Part 121 aircraft is climbing out at 3000 ft/min from a nearby airport and encounters the cruising GA aircraft. (Basis: ASRS, 4%)	RVV: + 3000fpm RHV: 0 - 1200kts IA: any
A6	A Part 121 aircraft is on and IFR approach to an airport. A GA aircraft is climbing out via the standard airport pattern having recently departed from the same airport. The two aircraft encounter each other. (Basis: ASRS, 5%)	RVV: > 3000fpm RHV: 0 - 1200kts IA: any

Figure 3-17: Narrative, Location and Geometry of Encounter Category A5 and A6

A7	A fixed wing aircraft is on final to a non-towered airport. A helicopter is hovering next to the runway. The fixed wing aircraft performs a touch and go. Upon touchdown of the fixed wing aircraft, the helicopter departs, expecting the aircraft to make a full stop landing. The two aircraft encounter each other on upwind. (NTSB, 1 case)	RVV: < 200 fpm RHV: < 50 kts IA: any
A8	A fixed wing aircraft is on final to a non-towered airport. A helicopter is also approaching the airport but does not join the standard airport pattern. The two encounter each other on short final. (Basis: NTSB, 1 case, e.g., practice auto-rotations)	RVV: < 200 fpm RHV: < 200 kts IA: any

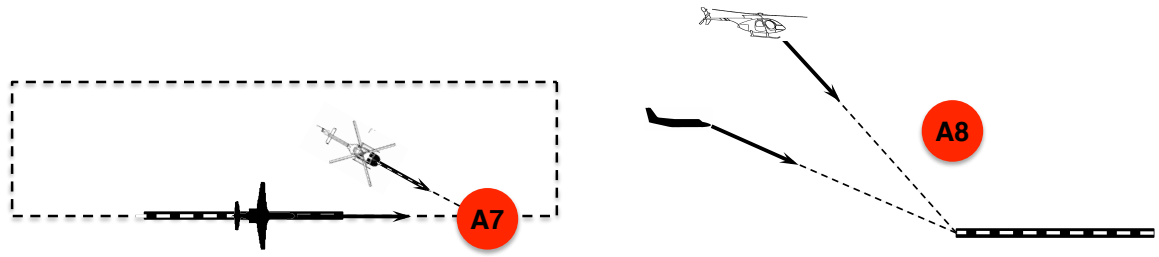


Figure 3-18: Narrative, Location and Geometry of Encounter Category A7 and A8

E1	One aircraft is performing maneuvers (such as circling, flight training, hovering) while another aircraft is transitioning through the same area. The second aircraft is flying straight but may be level or climbing/descending. (Basis: NTSB, 10.5%)	RVV: 0 - 3000fpm RHV: 0 - 350kts IA: any
E2	Both aircraft are conducting maneuvers. An example of this would be two news helicopters operating in the same area or a firefighting operation. (Basis: NTSB, 4.2%)	RVV: 0 - 3000fpm RHV: 0 - 200kts IA: any

E3	An IFR General Aviation aircraft is cruising at 10,000 ft transitioning past a Class C airport. An arriving Part 121 aircraft is descending at 3000 ft/min to the primary airport with in the Class C airspace and encounters the cruising GA aircraft. (Basis: ASRS, 7%)	RVV: 0 - 3000fpm RHV: 0 - 1200kts IA: any

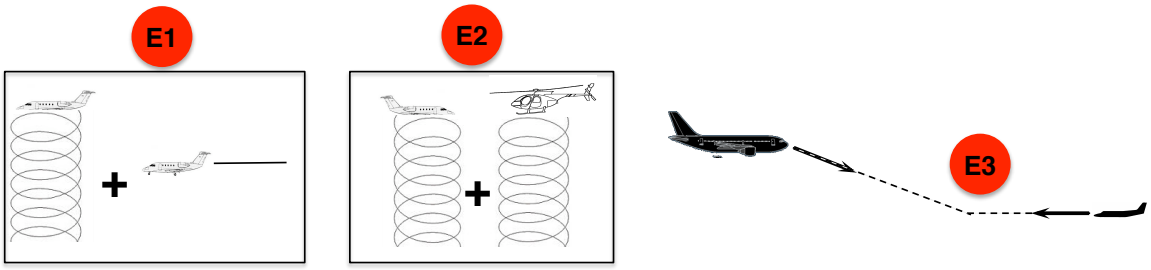


Figure 3-19: Narrative, Location and Geometry of Encounter Category E1, E2 and E3

E4	Two aircraft converge on each other en-route with an angle of less or equal than 60° between their tracks. Both aircraft may be climbing or descending. (Basis: NTSB, 11.6%)	RVV: < 3000fpm RHV: 0 - 300kts IA: < 60°
E5	Two aircraft converge on each other en-route with an angle of more than 60 but less than or equal to 120° between their tracks. Both aircraft may be climbing or descending. (Basis: NTSB, 10.5%)	RVV: < 3000fpm RHV: 0 - 1040kts IA: 61 - 120°
E6	Two aircraft converge on each other en-route with an angle of more than 120° between their tracks. Both aircraft may be climbing or descending. (Basis: NTSB, 9.5%)	RVV: < 3000fpm RHV: 0 - 1200kts IA: > 120°

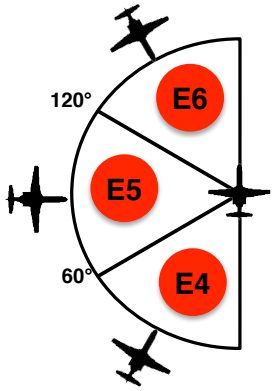


Figure 3-20: Narrative, Location and Geometry of Encounter Category E4, E5 and E6

3.3 Identification of Architectural Requirements for TSAA

When installed, TSAA will be one of many systems that operate on-board an aircraft. This fact has two important implications. First, TSAA must be designed to meet standards that define the system architectures and interfaces for avionics on-board aircraft. Second, TSAA must be capable of operating alongside other, previously certified systems without contradicting them or interfering with their functions. Each of these implications results in system requirements and is addressed individually below.

3.3.1 Interface Definitions

The DO-317A standard defines the avionics architecture and data interfaces for systems that use ADS-B. Effectively, the standard outlines the method by which the various sources of surveillance and operational data are to be combined on-board the own-ship and how that data are to be processed before passing on to other systems (see Figure 3-21). Known as the Airborne Surveillance and Separation Assurance Processor (ASSAP), it contains a tracker that receives data via ADS-B In (i.e., air-to-air ADS-B, ADS-R and TIS-B), active surveillance sensors (such as those used for TCAS systems), and any other data source. Based on the received data, the tracker then generates a single set of states for any surveyed target.

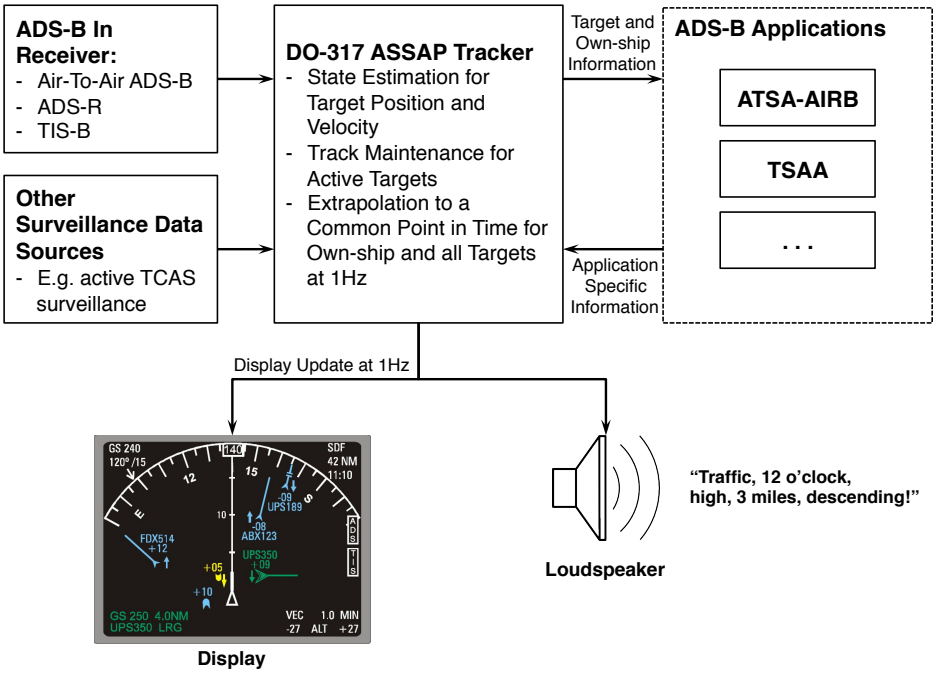


Figure 3-21: Notional Avionics Architecture with DO-317A ASSAP Processor

At a minimum, the DO-317A standard requires the tracker to take the raw ADS-B data received from all targets in the vicinity, select the best source to be used if more than one data source is available for a particular target, and then provide three dimensional position and velocity vectors at 1Hz for each target. If the available data from the own-ship and the targets differ in their time of reception, the tracker must also extrapolate position and velocity to a common point in time (CPT).

The DO-317A standard contains the code for a sample implementation of the ASSAP tracker that, if implemented by a manufacturer, meets the DO-317A standard performance requirements. The sample tracker uses three independent, two-state extended Kalman filters to reduce the effect of potential surveillance errors that may be present in the received data and assembles a "best-estimate" track for each target for which data are available. However, a manufacturer is not required to implement the sample tracker and may decide instead to use a different approach as long as it achieves the standard's performance requirements. The sample tracker also contains more functionality than what is required as a minimum by the DO-317A standard. As a result, not only may the approach used by the manufacturer be different than the one proposed by the sample tracker, but the data may have undergone more or less conditioning by the time it is received by TSAA.

Table 3-4: State Data Available to TSAA from DO-317A Tracker (According to Table H-2 in [25]**)**

State Report Element
Target ID
Aircraft Type
Report time of Applicability
Reported Latitude and Longitude
Estimated Latitude and Longitude
Geometric Altitude (above WGS-84 Geoid)
Barometric Altitude
Reported North/South Velocity
Reported East/West Velocity
Estimated North/South Velocity
Estimated East/West Velocity
Reported Vertical Rate
Estimated Vertical Rate

From a system requirements perspective, TSAA should be designed to operate with a DO-317A tracker with Figure 3-21 and Table 3-4 defining the high-level system interfaces. However, from an analysis perspective it is unclear what assumptions should be made about how conditioned the surveillance data will be by the time it is received by TSAA.

Therefore, moving forward, two different implementation architectures of the TSAA algorithm will be of importance. In the first, TSAA is implemented as a standalone algorithm that receives the raw ADS-B data. This implementation represents the case where the data received by TSAA has undergone no conditioning at all. In a real-world implementation this will likely never be the case but for the purposes of analysis it represents a worst-case scenario. A schematic of this implementation is shown in Figure 3-22. As shown, since there is no tracker to extrapolate all tracks to a common point in time (CPT), that functionality has to be performed by TSAA, along with the estimation any other states that are not provided by ADS-B (e.g., turn rate).

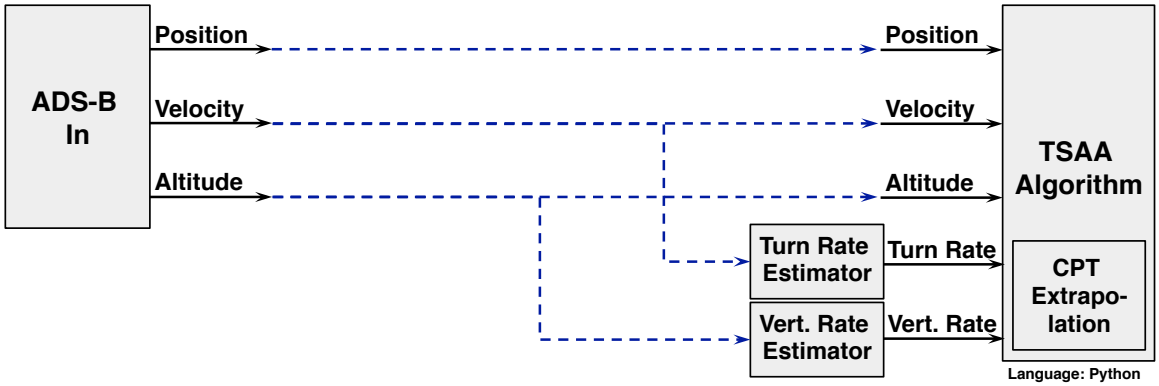


Figure 3-22: Visualization of the Stand-Alone TSAA Implementation (no DO-317A Tracker)

In the second implementation architecture, the algorithm is implemented in conjunction with the sample tracker described in the DO-317A standard. Compared to the stand-alone implementation, this implementation represents the case wherein the data has undergone significant conditioning by the tracker. Figure 3-23 shows a schematic representation of the implementation architecture of TSAA with a DO-317A tracker. While the previous implementation was of importance from an analytical perspective, this implementation is of importance from a design and interoperability perspective. In this implementation, any states required by TSAA that are not provided by the tracker must be estimated independently.

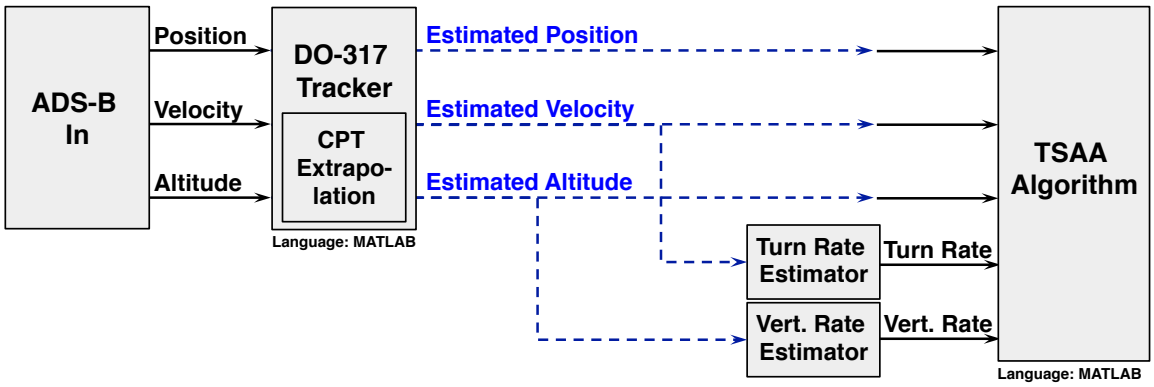


Figure 3-23:TSAA Implementation With A DO-317A Tracker

It should be noted that, given the same input data, TSAA's performance is expected to be better when implemented with a DO-317A tracker than without one. This is due to the fact that the tracker effectively removes a portion of the noise that may be present in the information received via ADS-B. Without a tracker, that noise would be fed directly into the TSAA algorithm.

Aside from the requirement for TSAA to operate with the DO-317A tracker, an additional, subtler system requirement flows from the architecture shown in Figure 3-21. Compared to current alerting systems, the state data that TSAA ultimately uses may originate from a variety of sensors (e.g., a ground based radar for TIS-B targets, a GPS system for ADS-B targets, etc.). Compared to TCAS systems that use a single sensor with well-known performance and uncertainty characteristics, ADS-B data may be derived from a range of sensors, which causes the associated current state uncertainty to vary from target to target. By extension, TSAA must have the capability to maintain alerting performance across a range of uncertainty levels in current state information.

Lastly, as mentioned in section 2.3, a precedent for two classes of alerting systems already exists in the current standards for TAS. This precedent has two significant benefits. First, it supports the requirement of keeping costs low by not requiring a horizontal situation display and issuing aural alerts only. Second, it has the benefit that aircraft that either do not have the necessary panel space for a display or for which the installation of a display is undesirable could still equip with TSAA and ADS-B, possibly increasing overall equipage in GA. Therefore, two classes of TSAA equipment should also be defined in the standard.

In summary, the following two system requirements can be defined with respect to system interfaces:

1. TSAA shall be capable of operating within the context of a DO-317A avionics architecture
2. TSAA system performance shall be evaluated using an implementation without a DO-317A tracker as a conservative approach
3. TSAA shall be capable of maintaining alerting performance across a range of levels in target state uncertainty
4. TSAA shall have to equipment classes, similar to the TAS equipment classes

3.3.2 Conformity To Previous Standards and Interoperability With Pre-Existing Systems

As an ADS-B application, TSAA must conform to previously defined operational procedures and regulations (e.g., ATC) and ADS-B standards as well as be compatible with currently operational systems (e.g., TCAS). As mentioned, DO-317A outlines the "behind the scenes" avionics architecture. It also defines a set of basic ADS-B applications and how they are to interact with the flight crew. One of those applications is particularly important to TSAA: the Air Traffic Situation Awareness – Airborne (ATSA-AIRB) ADS-B Application. ATSA-AIRB defines how ADS-B target information is displayed to the flight crew via a Cockpit Display of Traffic Information (CDTI; Figure 3-21), using the symbols shown in Figure 3-24. Additionally, ATSA-AIRB defines the minimum required quality of target data: the target horizontal position must be known with an accuracy of better than 0.5NM (NACp of 5) and the velocity must be known with accuracy better than 10m/s (NACv of 1) [87].

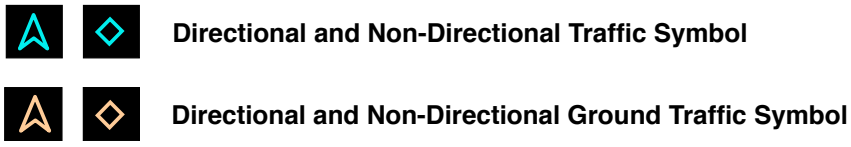


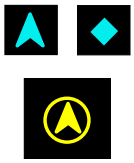
Figure 3-24: Traffic Symbols Defined in DO-317A for ATSA-AIRB

As an alerting application, TSAA builds on ATSA-ARIB by adding alerting to the basic functionality that ATSA-AIRB provides. By extension, TSAA systems must display all traffic using the same symbols as the ATSA-AIRB to maintain consistency between the two applications. However, since TSAA introduces an alerting functionality, additional symbols to differentiate alerted traffic from non-threat traffic must be introduced. Those symbols are defined in DO-317A and are shown in Figure 3-25.

The precedent for symbols of traffic that pose a potential threat to the own-ship comes from TCAS. The first symbol is for "proximate traffic" and is a solid version of the basic cyan symbol shown in Figure 3-24. A target is designated as proximate traffic if it is within 1200

ft vertically and 6 nautical miles horizontally of the own-ship [25]. A proximate target may never receive an alert; additionally, for an alert to be received on a target, the target does not first have to be designated as proximate traffic.

The second symbol is for traffic that has received a caution-level alert, requiring immediate awareness and subsequent action. As with traffic advisories (TAs) issued by TCAS I, caution-level alerts are associated with the color yellow.[8] Figure 3-25 only shows the caution level symbol for traffic with directional information. If no directional information is available the traffic is shown as a filled yellow circle.



Directional and Non-Directional Proximate Traffic Symbol

Directional Alert Traffic Symbol – Caution Level
(DO-317 does not define a Non-Directional Alert Traffic Symbol)

Figure 3-25: Proximate Traffic and Alerted Traffic Symbols Defined by DO-317A

An additional observation is that there are no subdivisions within the caution or the avoidance alerts. In other words, a system cannot issue alerts that differentiate between levels of caution. As such, TSAA cannot issue low-level caution alerts that, if the threat persists, escalate to mid- or high-level caution alerts.

In summary, TSAA as an alerting system will need to use the symbols and applications defined in the DO-317A standard to depict the alerts to the flight crew if a display is used. This results in the following two additional architectural system requirements:

5. TSAA shall be subject to the operational, display, and performance requirements defined for ATSA-AIRB
6. TSAA shall follow the currently existing guidance on symbol coloring for caution and warning level alerts, conform to operational procedures and regulations and be compatible with currently operational systems

[8] When a TCAS II system issues a resolution advisory (RA), the color of the traffic symbol is red. The color red is reserved for warning-level alerts, such as RAs, which generally are accompanied with executive guidance to the flight crew and require immediate awareness and immediate action. The use of yellow and red is also consistent across other, non-traffic alerting systems such as the enhanced ground proximity warning system (EGPWS).

3.4 Identification of Performance Requirements for TSAA

The primary performance metric of how well a conflict alerting system performs is whether it decides when to issue an alert correctly. As discussed in Chapter 2, this decision is made when the system predicts the presence of a hazard. In the case of conflict alerting systems, the hazard is a mid-air collision and is shown notionally as a white and red region in Figure 3-26. If an alerting system had access to perfect state information, a mid-air collision could be predicted perfectly and avoided every time. However, alerting systems only have access to limited state information; additionally, current state uncertainty and future state uncertainty limits how well a system can predict mid-air collisions. As a result, buffers around the hazard are included during the design of the alerting system to provide protection against those uncertainties; the orange alert region in Figure 3-26 represents this. Once in operation, an alerting system issues an alert when it predicts a violation of the alert region.

In the case of aviation, two aircraft are considered to have been involved in a near-mid-air collision if their slant range is less than 500 ft at any point during an encounter. Therefore, in order to ensure that an alerting system alerts not only on when a mid-air collision is predicted but also when two aircraft are expected to be in very close proximity, an additional buffer is included and the alert region further increased.

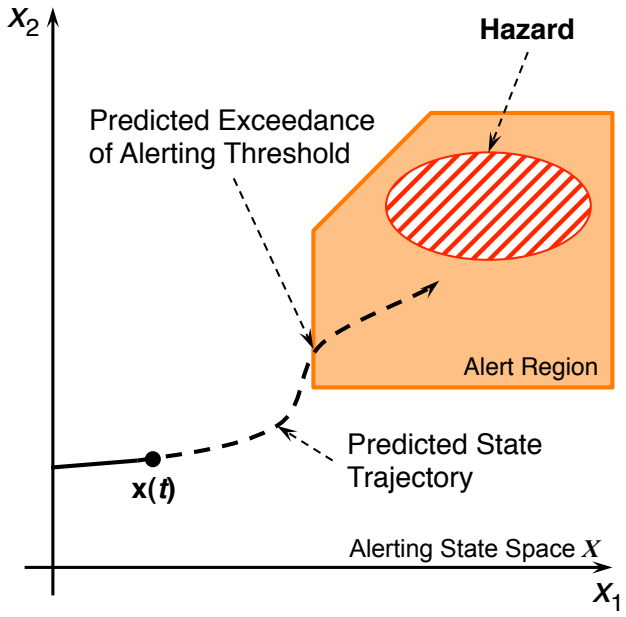


Figure 3-26: State Space Representation of Alerting Problem

ROLE OF ALERT PERCEPTION BY THE FLIGHT CREW IN ALERTING SYSTEM PERFORMANCE

When a flight crew receives an alert during operations, it evaluates the situation that generated the alert. Though the system has predicted that the hazard will be present, the flight crew's perception of the situation and whether or not a hazard is truly present may be different and depends on a variety of factors, including:

- **Operational Experience:** The overall level of experience of the flight crew or their familiarity with a particular operation.
- **Situation Awareness of the Flight Crew:** For example, if the flight crew had previous acquired a hazard and deemed it as "No Threat", an alert issued after the fact may be considered unnecessary, possibly even a nuisance. Additionally, if there is a threat but it is never acquired by the flight crew, the alert may appear to have been issued spuriously as a result of a system malfunction and result in no response by the flight crew [88]. A subtler version of this factor is the fact that the alerting systems is not aware of what knowledge the flight crew has already received, or whether the flight crew assimilated that knowledge correctly.
- **Operational Environment:** Certain events considered hazardous in the en-route environment may not be considered hazardous in the terminal environment.
- **Flight Phase:** During certain flight phases, the aircraft configuration may prevent the flight crew from effectively responding to an alert. For example, a "Climb, Climb" advisory to an aircraft on short final configured for landing and close to stall speed is effectively useless. This is in part the reason for the suppression of TCAS RAs below 700 ft AGL.
- **Overall Alerting Frequency:** Generally, the higher the overall rate of alerting becomes, the less tolerant flight crews are of incorrect alerts [89]. Systems that have high rates of alerts are frequently deemed "chatty" and are sometimes even switched off. On the other hand, the less a system alerts, the more tolerant flight crews may be of incorrect alerts as it confirms that the system is still operating, maintaining their trust.

A case of special interest is when the alerting system's perception differs from the flight crew's perception of whether or not a hazard is present in the system. An alert issued in this case can be perceived in a variety of ways. On one end of the spectrum, the flight crew may perceive the alert as a correct alert, understand why the system would consider the situation hazardous, but deem the alerted aircraft as no threat to the own-ship. In this case, the alert may in fact increase flight crew's trust in the system [2], [89]. On the other end of

the spectrum, the flight crew may not perceive a hazard to be present and thus consider any alert to be a nuisance.

Therefore, two main factors affect the sizing of the alert region in Figure 3-26: uncertainty in current and future states and the subjective definition of what constitutes the presence of a hazard.

Given these observations, alerts are separated into three groups at a conceptual level:

1. **Correct Alerts:** Alerts that were issued when a hazard was present or that the flight crew considers warranted given the current situation
2. **Nuisance Alerts:** Alerts that were issued when no hazard was present in the system and the flight crew perceives the alert as unwarranted
3. **Missed Alerts:** Situations where no alert was issued but a hazard was present

It is important to note that in the case of TCAS-like systems that use an active surveillance system for target surveillance, the alerting thresholds had to be set conservatively to account for the state uncertainties introduced by the limitations of the surveillance sensor. As a result, the size of the alerting region was larger than if it had been set according to what constituted the subjective presence of a hazard in some environments (e.g., airport environment). Since ADS-B can have significantly less uncertainty in its state information, the sizing of the alerting region is no longer limited by this uncertainty and can be tailored more specifically to what flight crews perceive to be the presence of a hazard.

3.4.2 Definition Of a Performance Standard For Alerting System Evaluation

During the design of the alerting system, the manufacturer of the system defines the size of the alert region in Figure 3-26. Presumably, different manufacturers will use different approaches to determining when alerts are appropriate, which will result in different alerting behaviors. Therefore, an objective and independent method is needed to evaluate how successful a given system is at detecting the presence of a hazard and at alerting the flight crew in the same situations in which the flight crew would identify the alerted aircraft as a hazard themselves. Such a method was developed during the design of TSAA and is presented here.

The method uses three own-ship centric zones that define a required alerting behavior if a target enters into them. The three zones are listed below and visualized in Figure 3-27.

- **Hazard Zone:** If an aircraft penetrates this zone, the hazard is considered to be present, and therefore an alert should be issued

- **Non-Hazard Zone:** If an aircraft remains in this zone, no hazard is considered to be present and an alert is therefore undesirable
- **May Alert Zone:** If an aircraft penetrates this zone, a hazard may or may not be present and an alert may or may not be issued

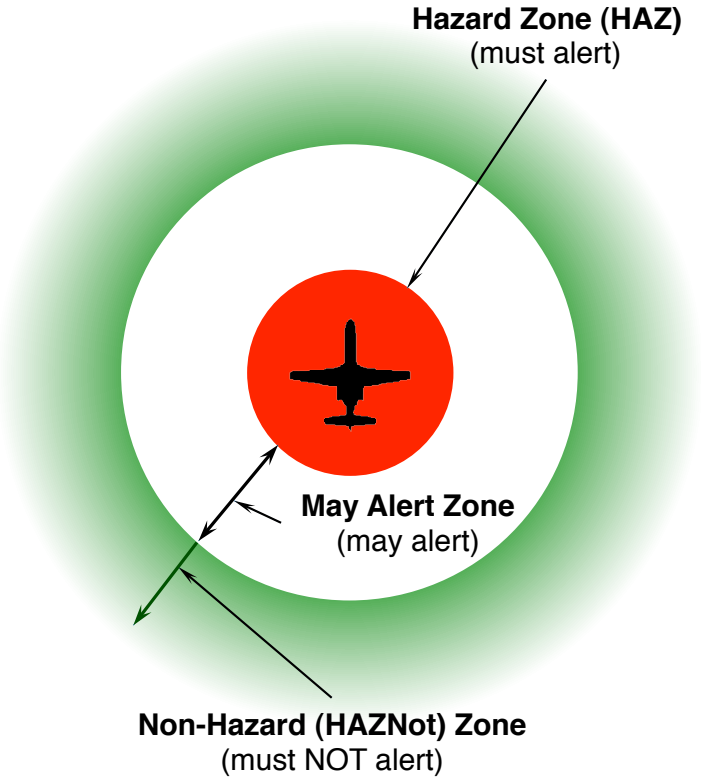


Figure 3-27: Zones Used in Alert Evaluation

In order to properly size the zones given the definitions above, MITRE acting on behalf of the FAA presented this scoring approach to a group of pilots from the US and Europe. The group consisted of 24 pilots with logged flight times ranging from 250 hours to 33,000 hours and certifications ranging from Private Pilot to ATP. On average, the pilots had 790 hours of flight time experience with TCAS I type systems (TAS, TIS, etc.) and an average of 2750 flight hours in aircraft equipped with TCAS II.

Based on their operational experience, the group was asked to determine the shape and size of the zones. Given the discussions above about the perception of hazards for an operational environment and encounter situation, the zones were sized specific to three operational environments, as shown in Table 3-5. Additionally, the shape of the zones was selected to be

circular based on a consensus of the group. Other shapes such as elliptical zones or zones shaped like spades extending further in the direction of flight were considered to capture differing levels of comfort between head-on and in-trail encounters. However, it was unclear which shape would be most appropriate or what its dimensions and ratios should be – as a result, circular zones represented the most attractive compromise.

Table 3-5: Size of Zones used for Alert Evaluation

	Hazard Zone		Non-Hazard Zone	
Environment	**Vertical**	**Horizontal**	**Vertical**	**Horizontal**
Terminal	200ft	500ft	500ft	1/2NM
En-Route (< 10,000MSL)	450ft	500ft	850ft	2NM
En-Route (> 10,000MSL)	450ft	500ft	850ft	3NM

Using this scoring method, formal definitions of how to score a particular alert issued by an alerting system can be introduced.

3.4.3 DEFINITION OF TECHNICAL PERFORMANCE METRICS FOR TSAA

Given the flight tracks of two aircraft and what alerts were issued, the following approach will be used to score the performance of TSAA. First, the system calculates the point along the trajectories where the slant range is the smallest and denotes this as the closest point of approach (CPA). Note that the CPA is defined here as the closest point of approach between two aircraft that are not aware of each other – as a result, no actions are taken by the flight crews to alter the flight path due to the presence of the other aircraft. The CPA of the track pair and any alerts that may have been issued during the encounter are then compared against the zones defined above.

The process described above is for a single flight track. However, in order to understand the aggregate performance of the alerting system with statistical significance, the process is repeated for a large data set of encounters. This allows for the calculation of the following technical performance metrics:

- **Nuisance Alert Rate:** A nuisance alert is any alert that is issued during an encounter where the CPA between the flight tracks remains in the Non-Hazard Zone. The nuisance alert rate is defined as nuisance alerts issued per own-ship flight hour.
- **Missed Alert Percentage:** A missed alert is defined as an encounter where the CPA falls within the Hazard Zone but no alert is issued by the alerting

system. The missed alert percentage is defined as missed alerts expressed as a percentage of all encounters with a CPA in the Hazard Zone.

- **Late Alert Percentage:** Given an encounter with a CPA that lies within the Hazard Zone, a late alert is defined as an alert issued with less than 12.5 seconds before the CPA. 12.5 seconds has been identified as the time required for a flight crew to receive an alert, acquire the threat, determine the best course of action and execute that action [90].
- **Average Time of Alert Before CPA:** The average time of alert before CPA is defined as the time before CPA averaged across all non-nuisance alerts.
- **Total Alert Count**: The total alerts issued for a given set of encounters.

In addition to the sizes of the zones, the focus group was also asked to set levels of acceptable performance rates for nuisance, late, and missed alerts. As discussed in the next paragraphs, an alert issued too late to allow for the flight crew to respond to the situation can have a similar effect as a missed alert. As a result, late alerts were combined with missed alerts. For both, the group defined a desirable performance level of no greater than 5%[9].

Table 3-6: Acceptable Performance Levels as Defined by the Pilot Focus Group

Type of Alert	Desired Performance Level
Nuisance Alert	5%
Missed Alerts + Late Alerts	5%

Given this independent scoring method, any alerting system can be evaluated objectively and its performance measured independently of alerting system implementation, which allows for direct comparison of specific performance metrics between multiple alerting systems. In the case of TSAA, this will enable its performance to be compared against a TCAS-like system, since the basic TCAS algorithm used in most current alerting systems serves as a benchmark against which TSAA should be compared. As such, an implied TSAA performance requirement is that the performance of TSAA shall be better than that of a current system based on the TCAS algorithm.

[9] Given that the rate of nuisance alerts depends on the operational environment, the group defined the percentage of nuisance alerts instead of a rate. During the evaluation of a particular environment, this percentage can then be translated into a rate, given the total number of flight hours in the data set and the number of alerts issued by the alerting system.

A couple of observations must be made at this point. First, the approach outlined to identify nuisance alerts technically can identify *unnecessary* alerts only; whether the alert is in fact a nuisance depends on how it is perceived and is subject to the additional factors as described on page 76. However, since such alerts are most commonly referred to as nuisance alerts in the context of conflict alerting systems, this term will also be used here.

Second, a distinction between nuisance alerts and false alerts must be made. The term false alert is considered here to be an alert that is issued as a result of a physical malfunction of the alerting system. In other words, an alert is only scored as a nuisance alert if the system is operating as intended, without any hardware or software faults.

Third, since alerting systems do not issue executive commands on how to avoid a threat, ensuring that an alert is issued with enough time for the flight crew to respond becomes more important. As a guideline, 12.5 seconds has been identified as the minimum amount of time a pilot needs to understand an alert, acquire the target, decide on an action, and execute that action [90].[10] As a result, during the development of TSAA, alerts issued with less than 12.5 seconds before the CPA on encounters that penetrate the Hazard Zone are considered late and counted as part of the missed alert percentage.

Fourth, a possible limitation of the results from the focus group is that its members reported experience with almost exclusively TCAS-like systems and the associated overall alert rates. Given this experience, the mental model of the group members may have influenced the selection of the overall nuisance alert percentage; for example, higher nuisance alert percentages may be acceptable for alerting systems that alert less frequently than TCAS-like systems overall. As a result, this nuisance alert requirement can more appropriately be recalculated as a nuisance alert rate in terms of hours between nuisance alerts, based on the overall alert rate of the TCAS-type system. This number can then be compared directly against alerting systems with different overall alert rates.

Fifth, regarding missed alerts, systems frequently are constructed with a bias to prevent missed alerts at a cost of introducing higher levels of nuisance alerts, for obvious reasons [91], [92]. However, an additional factor that can potentially influence how alerting systems are designed is the safety and design assurance level requirements that must be met for certification. The higher the criticality of the system, the more stringent its failure conditions become. As outlined in DO-178B, the maximum probability of failure per flight hour allowed decreases from 10^{0} to 10^{-9} as the severity of such a failure increases from

[10] By comparison, TCAS II avoidance systems that do give executive commands to the flight crew assume only 5 seconds of necessary response time.

Minor to Catastrophic. Therefore, during the design and development phase of some systems the probability of failure threshold must be achieved, and this often occurs at a cost of higher levels of nuisance alerts.

Lastly, regarding CPAs, aside from the definition used here, two alternative definitions of CPA definitions are frequently of interest. First, the CPA predicted by the alerting system during the encounter gives insight into how well the alerting system prediction matches how the encounter truly unfolds. Second, in cases where the flight crew responds to an alert, the CPA is of interest and can be used as a measure for evaluating the effectiveness of an alerting system. If an alert is issued during an encounter and the flight crew responds to it, the CPA will be different than if the flight crew had not responded. Frequently, the separation at the CPA will be larger than it would have been in the absence of an alert. However, when scoring such an encounter, a larger CPA can fall into the Non-Hazard Zone, resulting in an alert being scored as a nuisance alert when in fact the system was working as desired. Therefore, in order to calculate the performance metrics defined above, the data set of encounters used during the alerting system evaluation is assumed to consist of encounters where the two aircraft are not aware of each other.

In summary, the following performance requirements have been identified for TSAA:

1. The performance of TSAA shall exceed the performance of a TCAS I or TAS-like system
2. The missed alert percentage of TSAA shall not be larger than 5%
3. The nuisance alert percentage of TSAA shall not be larger than 5%

3.5 Considerations on Potential Interactions between TSAA and Collision Avoidance Systems

One consideration of particular interest for TSAA is its potential to interact with avoidance systems such as TCAS II. As discussed above, when two aircraft equipped with an avoidance system such as TCAS II encounter each other, the two systems coordinate the avoidance commands (i.e., the RAs) before they are issued in order prevent giving a combination of commands that might lead to prolonged high-risk exposure. In an encounter wherein only one aircraft is equipped with a collision avoidance system, this coordination does not take place. Instead, the RA is issued based on the assumption that the threat aircraft will continue operating as before.

This second scenario is of concern during the design of an alerting system. Specifically, if an aircraft equipped with an *avoidance* system encounters an aircraft with an *alerting* system,

there is the possibility that an alert by the alerting system will cause the flight crew to maneuver in a way that the avoidance system does not expect. Since the avoidance commands against non-coordinated targets are issued based on the assumption that the target aircraft is not going to maneuver, any maneuver on behalf of the target aircraft has the potential to aggravate the situation or prevent the issued command from effectively resolving the situation.[11] It should also be noted that this type of interaction might occur even in the absence of an alerting system (e.g., due to ATC traffic call-out or see-and-avoid activities by the flight crew).

In general, the effect of a system such as TSAA is that it increases the flight crew's situation awareness. Given this improved situation awareness, it is expected that the flight crew's response to an encounter would be more appropriate than without the system. Additionally, as later chapters will show, TSAA's alert rate is much lower than that of current alerting systems such as TCAS II. As a result, it is expected that the avoidance system will frequently issue its alerts without TSAA ever issuing an alert. This is effectively identical to the situation wherein an aircraft is not equipped with an alerting system to begin with.

In the case where both systems alert, any maneuver by the aircraft with the alerting system that is at odds with the avoidance command is to be avoided. Avoidance systems such as TCAS II do monitor the threat aircraft for the duration of the encounter and, if necessary, issue updated or reversed commands if the situation does not improve. Additionally, as mentioned above, given the improved situation awareness provided by a system like TSAA, a deliberate maneuver on behalf of the flight crew is highly unlikely. Nonetheless, a few observations can be made regarding how to reduce the likelihood of an undesirable maneuver.

MAXIMIZING AVERAGE ALERT TIME OF TSAA

One approach to reducing the odds of an undesired maneuver is to adjust the algorithmic approach used by TSAA. Ideally, TSAA would always alert before the avoidance system. In doing so, the flight crew on-board the aircraft with the alerting system would receive the alert and respond to it before the avoidance system on the other aircraft issues a command to flight crew. The response of the flight crew can then be taken into account before the avoidance system issues its avoidance command to the flight crew. Table 3-7 shows the alert time before CPA that would be required to ensure that maneuvers in response to an alert occur before the issuance of an RA by TCAS II, assuming a response time of 12.5

[11] Appendix B shows that this type of interaction has not recently lead to mid-air collisions or near mid-air collisions. Nonetheless, the potential for this interaction to exist warrants further analysis.

seconds. The tau values shown for TCAS represent the time to CPA under the assumption that the current range will continue to decrease at the current range-rate.

Table 3-7: Required Time of Alert Before CPA to Ensure Alerts are Issued Before TCAS II RAs

Own-ship Altitude (ft)	Time of RA Issuance by TCAS II (tau in sec)	Theoretically Required Time of TSAA Alert Issuance (+12.5 sec)
< 1000 (AGL)	N/A	N/A
1,000 – 2,350 (AGL)	15	27.5
2,350 – 5,000	20	32.5
5,000 – 10,000	25	37.5
10,000 – 20,000	30	42.5
> 20,000	35	47.5

Figure 3-28 shows the trade between nuisance alert rate and average time of alert before CPA for the exemplar TSAA algorithm introduced in the next chapter. As can be seen, if all alerts were required to be 47.5 seconds or earlier with respect to the CPA, the nuisance alert performance will be significantly worsened and may even be difficult to achieve. Therefore, forcing TSAA to always alert ahead of an avoidance system is impractical, although it may be a consideration during the design.

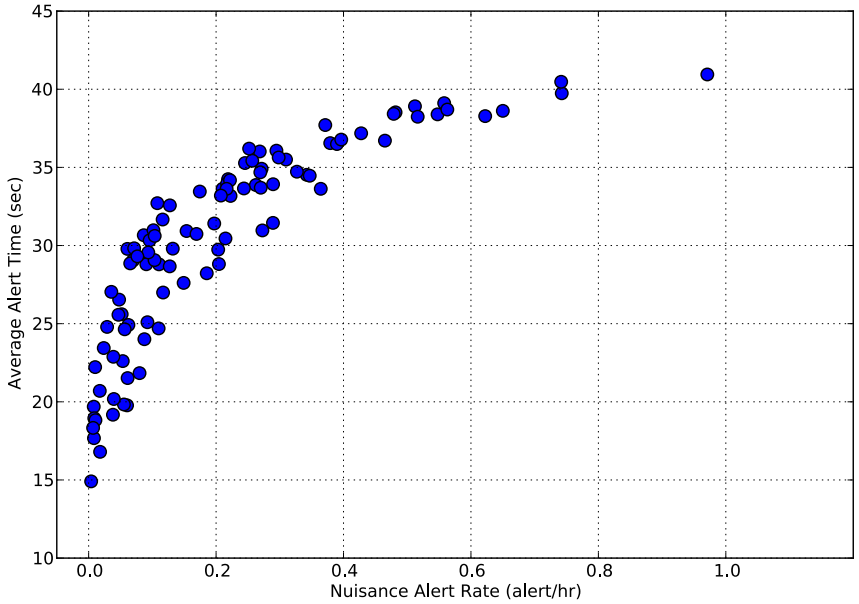


Figure 3-28: Nuisance Alert Rate vs. Average Time of Alert Before Closest Point of Approach

OPERATIONAL APPROACHES TO PREVENTING UNDESIRABLE MANEUVERS

As discussed in section 2.3, TCAS II systems only issue avoidance commands in the vertical dimension. Thus, a different approach to prevent undesirable maneuvers is to train pilots using a TSAA system to initiate horizontal maneuvers only in response to a TSAA alert. If the other aircraft is equipped with an avoidance system and a command is issued, the command will be in the vertical dimension, while the TSAA aircraft will maneuver in the horizontal dimension only.

TCAS II assumes that within 5 seconds of an avoidance command, the flight crew begins an initial acceleration of 0.25g to achieve a 1500 fpm climb or sink rate [51]. Comparatively, to achieve a similar relative motion in the horizontal dimension for commercial aircraft, a significant bank angle and/or time would be required. However, non-jet, GA aircraft frequently operate at lower speeds and routinely execute turns at higher bank angles.

Though this approach may mitigate potential issues during encounters between TSAA aircraft and aircraft equipped with avoidance systems, it introduces another set of concerns. First, as avoidance systems such as ACAS X are introduced and certified in the future, there is no guarantee that such systems will continue to issue commands in the vertical dimension only. Additionally, a horizontal maneuver may not always be the most prudent course of action. As described in FAR 91.113, the right-of-way rules dictate how two aircraft are to maneuver with respect to each other in situations where sufficient time is available. However, if a threat is imminent and an action is required immediately, a vertical maneuver or a combined horizontal and vertical maneuver may be more efficient.

As a result, if time permits, horizontal maneuvers according to FAR 91.113 are desirable during encounters with avoidance systems that issue avoidance commands in the vertical dimension. If time does not permit, it may become necessary for the TSAA-equipped aircraft to execute a vertical maneuver. The flight crew of aircraft equipped with TSAA systems should be made aware of this maneuvering trade-off through regulatory guidance or in training material that accompanies the TSAA installation.

3.6 Summary of High-Level System Requirements Identified for TSAA

The following high-level system requirements have been identified in this chapter. Also given here is a numbering scheme that will be used to refer to these system requirements throughout the rest of this thesis.

Table 3-8: Summary of Stakeholder Requirements

Requirement Number	Requirement
SH1	The equipage costs shall be kept to a minimum
SH2	TSAA shall be an alerting system and not an avoidance system
SH3	TSAA shall be able to alert during highly dynamic operations
SH4	TSAA shall be able to alert on all ADS-B targets, including TIS-B (radar derived) targets
SH5	The exemplar TSAA system be able to serve as the basis for the certification standard
SH6	The time to the introduction of TSAA shall be kept to a minimum
SH7	TSAA shall be primarily designed to operate with ADS-B targets

Table 3-9: Summary of Functional Requirements

Requirement Number	Requirement
FR1	TSAA shall reliably alert in all 14 scenario categories

Table 3-10: Summary of Architectural Requirements

Requirement Number	Requirement
AR1	TSAA shall be capable of operating within the context of a DO-317A avionics architecture
AR2	TSAA system performance shall be evaluated using an implementation without a DO-317A tracker as a conservative approach
AR3	TSAA shall be capable of maintaining alerting performance across a range of levels in target state uncertainty
AR4	TSAA shall have to equipment classes, similar to the TAS equipment classes
AR5	TSAA shall be subject to the operational, display and performance requirements defined for ATSA-AIRB
AR6	TSAA shall follow the currently existing guidance on symbol coloring for caution and warning level alerts, conform to operational procedures and regulations and be compatible with currently operational systems

Table 3-11: Summary of Performance Requirements

Requirement Number	Requirement
PF1	The performance of TSAA shall exceed the performance of a TCAS I or TAS-like system
PF2	The missed alert percentage of TSAA shall not be larger than 5%
PF3	The nuisance alert percentage of TSAA shall not be larger than 5%

Based on these system requirements, the TSAA exemplar algorithm was developed.

Chapter 4

DESIGN OF THE EXEMPLAR TSAA ALGORITHM

Given the system requirements derived in Chapter 3, a new alerting algorithm was designed for TSAA. The need to minimize algorithmic complexity and time to introduction (see stakeholder requirements SH5 and SH6), combined with the goal of ensuring that TSAA takes advantage of the additional data that ADS-B makes available, made designing a new algorithm the most attractive approach.

As discussed at length in Chapter 2, most current alerting systems use a TCAS-like algorithm. In the horizontal dimension, the TCAS algorithm operates with two states – namely range and range-rate – measured by active, transponder-dependent surveillance sensors on-board the own-ship. In the vertical dimension, the algorithm uses the discrete altitudes reported by the target's transponder and estimates a vertical rate using a non-linear tracker. Active surveillance systems can measure the azimuth to the target aircraft as well, but they do so with significant measurement error. As a result, the algorithm does not use azimuth when determining whether an aircraft is a threat to the own-ship. However, azimuth is used for a secondary filter to reduce nuisance alerts, as described in standard DO-197.

Since alerting systems based on the TCAS algorithm do not use azimuthal information to determine when to issue an alert, the horizontal geometry of the encounter is not taken into account directly; rather, geometry information is inferred indirectly via range, range-rate, and their ratio, which known as "tau". The tau is effectively the time to the closest point of approach under the assumption that the current range will continue to decrease at the current range-rate. As is obvious, this assumption is only true if no maneuvering occurs during this time. Once the tau falls below a certain threshold (refer to Table 3-7), the alert is issued.

Since a variety of different geometries can result in the same pair of range/range-rate values, alerting thresholds have to be set more conservatively in order to address the most dangerous of the geometries for a given range and range-rate. This conservative approach contributes significantly to the high rates of alerts issued by TCAS-like systems when operating in high-density environments such as the airport environment.

The state information that ADS-B makes available allows the encounter geometry to be determined in all three dimensions. This in turn potentially allows an alerting algorithm to make more informed decisions as to when an alert is appropriate, which would enable more precise alerting in environments such as the airport (system requirement FR1). Therefore, since the TCAS algorithm does not take geometry into account, it was not repurposed for TSAA.

In an effort to identify whether an algorithm proposed in the literature could be used for TSAA, the literature review summarized in Chapter 2 was conducted. Though some of the reviewed algorithms represented novel approaches to alerting (e.g., ACAS X), they were without precedent and tended to be complex, which was in conflict with the need for a low certification cost and time to achieve certification. Ultimately, the review identified no algorithm that could meet all the system requirements defined for TSAA.

In light of this observation, the design decision to develop a new algorithm for TSAA was made. The TSAA algorithm is based loosely on the example set by the TCAS algorithm but includes changes to the methods used for future state predictions and how the alerting thresholds are determined. More specifically, in addition to lower levels of current state uncertainty, the access to information about the geometry of the encounter allows future states to be predicted more accurately. This, in turn, allows for a more precise evaluation of whether a potentially hazardous situation may be present in the future.

4.1 Conceptual Introduction to the TSAA Exemplar Algorithm

Fundamentally, the concept of the exemplar TSAA algorithm is to predict discrete, constant turn-rate trajectories for all involved aircraft and alert the flight crew based on predicted penetrations of protected airspace along those trajectories. As such, the exemplar TSAA algorithm has three major components: protected airspace zones around target aircraft, trajectory prediction for own-ship and target aircraft, and alerting decision logic. A conceptual overview of each component is provided here; the next section focuses on the mathematical and software implementation of the algorithm.

The algorithm performs pair-wise evaluations to determine whether a conflict exists between the own-ship and a particular target. The algorithm calculates two protected airspace zones around each target; these are denoted as the protected airspace zone (PAZ) and the collision airspace zone (CAZ). Figure 4-1 shows the PAZ in yellow and the CAZ in red. The size of the PAZ depends on the closure rate between the target and the own-ship. As a surrogate for the potential danger involved in a given encounter geometry, the size of the PAZ increases as closure rate increases.

The size of the CAZ remains fixed at a radius of 500 ft and a height of ±200 ft, values that are based on the position uncertainty of two rule-compliant ADS-B targets. This definition agrees well with the definition that the pilot focus group provided for the presence of a hazard in the airport environment and, by extension, the sizing of the Hazard Zone (see section 3.4).

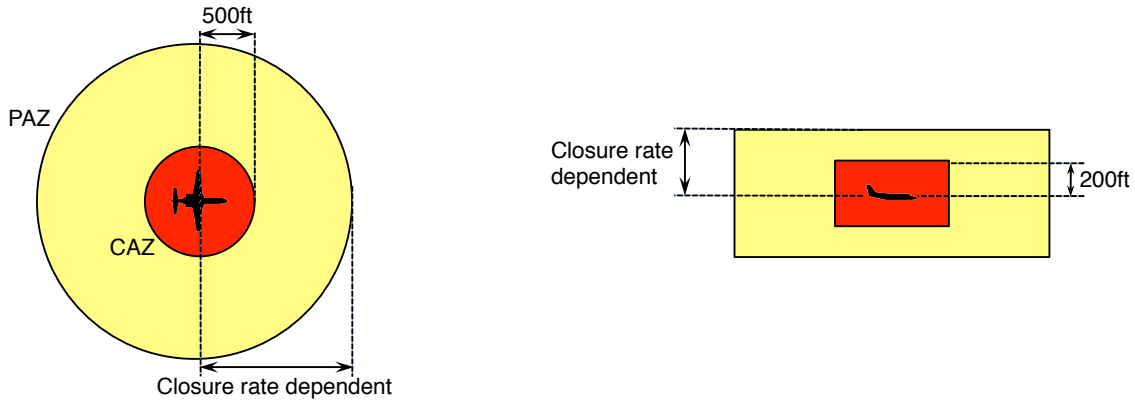


Figure 4-1: Schematic Representation of PAZ and CAZ Calculated by the Exemplar TSAA Algorithm

The second component of the algorithm is the trajectory prediction. For the own-ship as well as each target, discrete trajectories are predicted repeatedly at a nominal frequency (e.g., once per second). The TSAA algorithm uses a constant turn rate trajectory propagation; as such, the propagated trajectories predict where the aircraft will be if it continues its current maneuver. As shown in Figure 4-2, the constant turn rate prediction defaults to a constant heading prediction in the absence of maneuvering. During the initial design efforts, a constant turn rate trajectory projection was shown to reduce the nuisance alert rate while maintaining the desired level of missed alert percentages and average alert times.

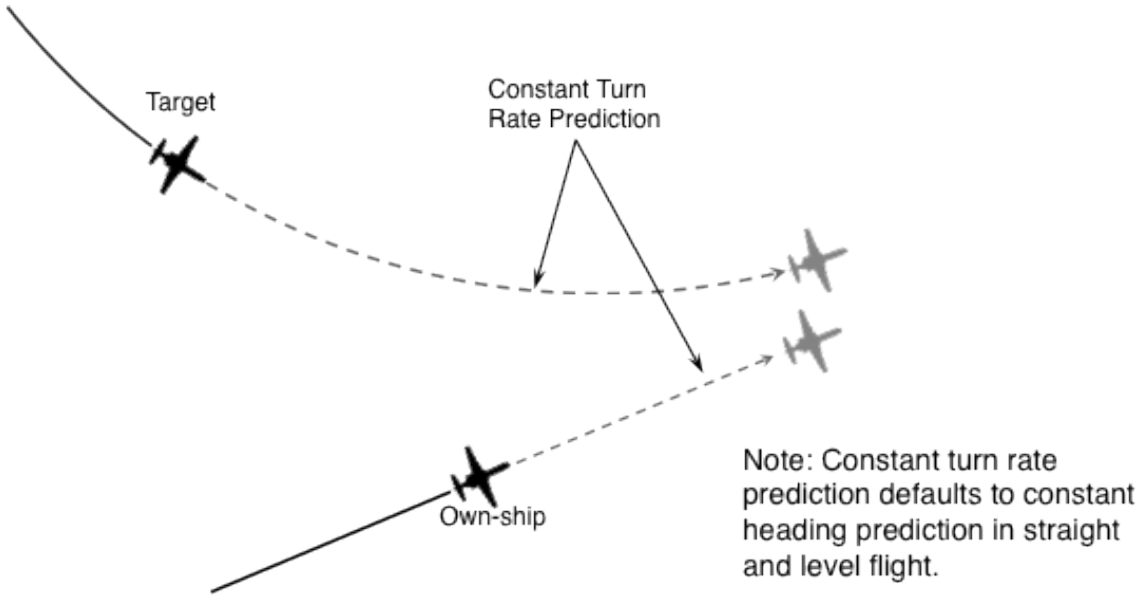


Figure 4-2: Constant Turn Rate Trajectory Projection

Since the geometry between the two aircraft can change along the trajectories due to the constant turn rate prediction, the closure rate and with it the size of the PAZ between the two aircraft can also change. For example, as shown in Figure 4-3, as the closure rate decreases along the trajectories, the size of the PAZ decreases.

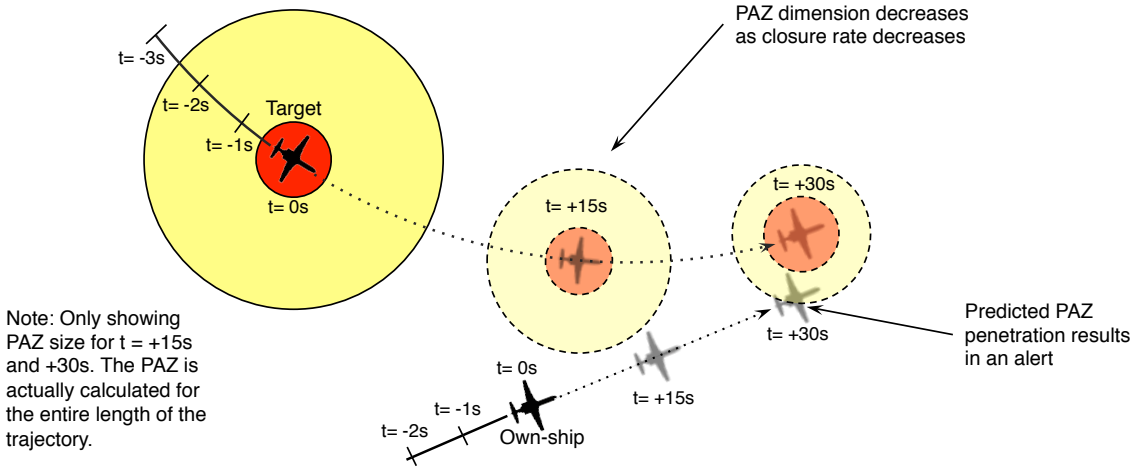


Figure 4-3: Schematic Representation of Alerting Logic Combining Protected Airspace Zones and Constant Turn Rate Trajectory Prediction

Based on the predicted positions of the aircraft and the sizes of the airspace buffer zones along the trajectory, the alerting logic determines whether or not to issue an alert for a given target. If the own-ship is predicted to penetrate the PAZ, an initial alert announcing the location and observed behavior of the target is issued to the pilot. An example alert would be "Traffic, Twelve O'clock, Three Miles, High, Descending". If the situation continues unchanged or deteriorates and the CAZ-penetration is predicted, an alert-*update* of the same format but with updated position and behavior information is issued to the pilot. In the example geometry shown in Figure 4-3, even though the PAZ decreases along the predicted trajectory, the own-ship is predicted to penetrate the PAZ 30 seconds into the future, which causes an initial alert to be issued to the flight crew.

As mentioned, the TSAA algorithm builds on the algorithmic approach used by the TCAS algorithm, and thus takes advantage of the TCAS I certification precedent. Specifically, by predicting discrete trajectories, the TSAA algorithm builds on the fact that TCAS I effectively predicts a constant heading trajectory when it evaluates range and range-rate. TSAA, however, adjusts the trajectory prediction to use constant turn rate to make it more applicable to highly dynamic environments. This approach has the benefit of approximating future states more accurately, which reduces the potential uncertainty associated with those future states. Additionally, TCAS I uses a protected airspace zone approach when defining the tau and distance values shown in Table 3-7. The TSAA algorithm extends that concept to high-density environments by allowing those protected airspace zones to adjust along the trajectory based on the geometry of the encounter.

4.2 Interface Definitions for the Exemplar TSAA Algorithm

The exemplar TSAA algorithm is designed with the intent of operating in the context of a DO-317A avionics architecture, as shown in Figure 3-21. As such, the data to and from TSAA and its interfaces are defined by the DO-317A standard. The tracker provides 3 major capabilities: the estimation of the received state information, the extrapolation of that information to a common point in time (in a common reference frame), and the provision of updated state estimates at a 1Hz update rate. Those major capabilities are assumed to be available for the TSAA implementation described below.[12]

[12] Note that as described in section 3.3, TSAA will also be evaluated without the tracker providing those capabilities. In this section, however, the capabilities are assumed to be present to simplify the description of the algorithm implementation.

It should be noted here that the scope of this interface discussion is limited to the algorithmic element of TSAA. As a larger system, TSAA also interacts with the flight crew when it issues aural alerts and visual indications of threat aircraft. This interface between the human and the larger TSAA system was the focus of a significant effort by Silva and Cho. Over two years, Silva and Cho conducted three human factors studies that determined the most efficient approach to pronouncing and displaying TSAA alerts to flight crews. The results of those studies have been published in a separate report [88].

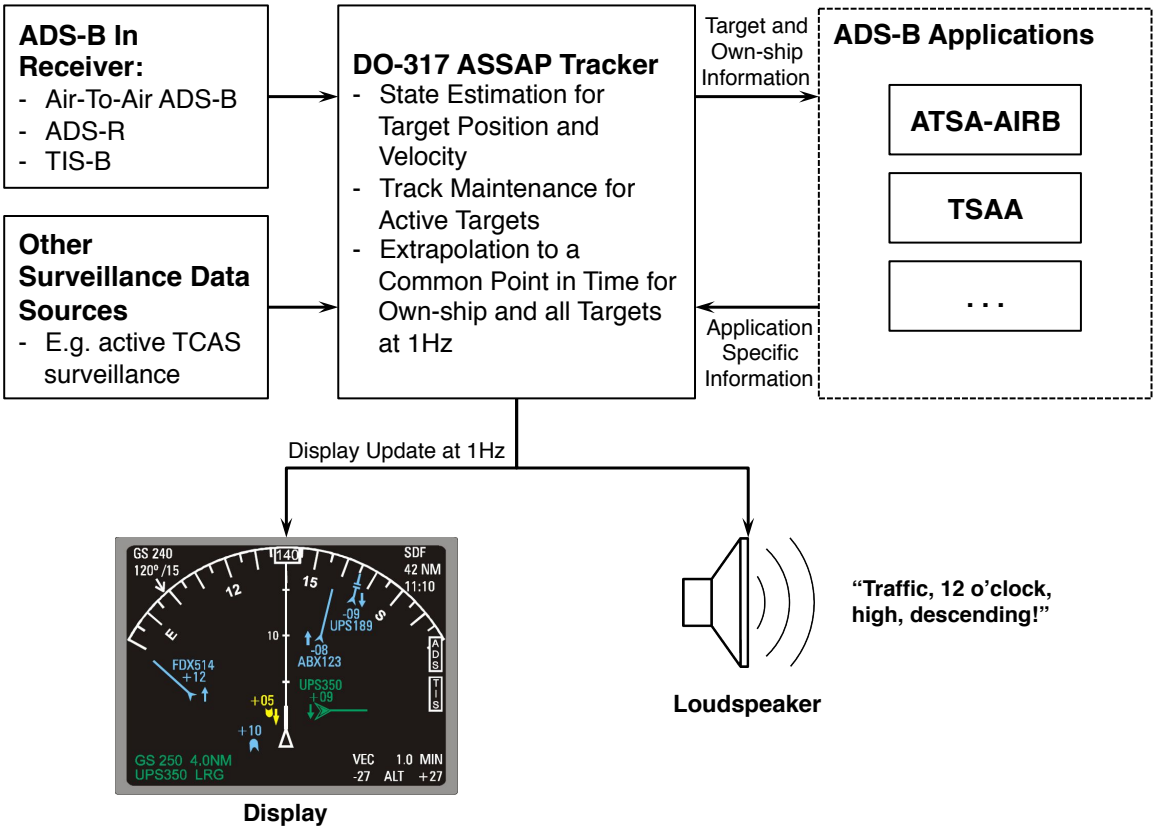


Figure 4-4: Notional Avionics Architecture with DO-317A ASSAP Processor

Figure 3-21 is reconfigured here as Figure 4-5 in order to represent the interaction between TSAA and the DO-317A tracker more specifically. Within TSAA, there are two major components, the Conflict Detector and the Threat Database, both of which will be described in more detail in the following sections.

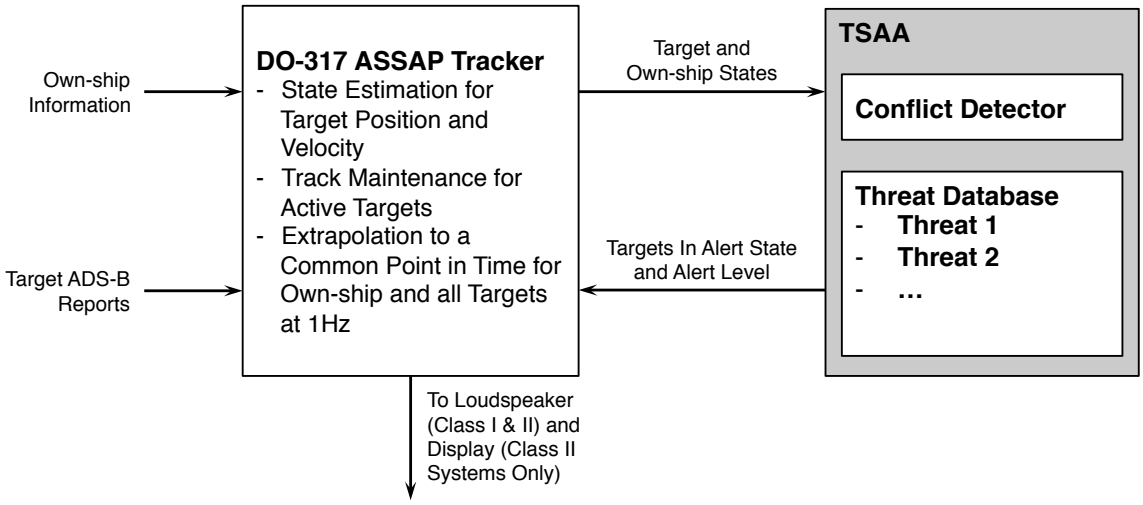


Figure 4-5: Notional DO-317A Avionics Architecture adapted for Implementation with TSAA

The information maintained by the tracker and passed to TSAA for a given target is listed in Table 4-1. As an output, TSAA provides a list of threat aircraft and their alert state (predicted PAZ violation or predicted CAZ violation). If a target that was passed to TSAA was determined to not be a threat, TSAA does not report it as an output.

Table 4-1: State Data Available to TSAA from DO-317A Tracker (According to Table H-2 in [25]**)**

State Report Element
Target ID
Aircraft Type
Report time of Applicability
Reported Latitude and Longitude
Estimated Latitude and Longitude
Geometric Altitude (above WGS-84 Geoid)
Barometric Altitude
Reported North/South Velocity
Reported East/West Velocity
Estimated North/South Velocity
Estimated East/West Velocity
Reported Vertical Rate
Estimated Vertical Rate

As can be seen, while turn rate is not a state maintained by the tracker, it is one of the states required by the TSAA algorithm. Additionally, the vertical rate maintained by the tracker is

the vertical rate reported via ADS-B. Representatives from the FAA advised the TSAA team not to rely on the reported vertical rate being accurate as current efforts are underway to evaluate possible limitations specific to older ADS-B implementations. Since both those states are necessary for the algorithm described above, TSAA will be required to estimate those states independently. As discussed in later sections, this is not an optimal solution and those functionalities should reside in the tracker. Nonetheless, the exemplar algorithm currently includes the functionality for estimating turn and vertical rate since the DO-317A tracker does not provide estimates of those states.[13]

It should be noted here that the algorithm as described in the following section assumes itself to be implemented in the context of the DO-317A tracker. However, as discussed in 3.3, from an analysis perspective an implementation without the benefits of a DO-317A tracker is of interest in order to determine the performance of TSAA in a worst-case implementation. To bridge the gaps in functionality that removing the tracker leaves, a coordinate system transformation from latitude and longitude to local coordinates and a constant point in time extrapolation function were added to the non-tracker implementation.

4.3 Mathematical Description of the Exemplar TSAA Algorithm

This section describes the exemplar TSAA algorithm mathematically. Appendix C contains a MATLAB implementation of the exemplar algorithm as it is described here. Thus, this description is specific to the implementation in the appendix.

As shown in Figure 4-5, TSAA has two main components: a Conflict Detector to identify threat aircraft and a Threat Database that maintains information about aircraft of interest. As a result, the TSAA exemplar algorithm can be called in two different modes:

1. **Update Mode:** When new state information is available from the DO-317A tracker, TSAA is called in this mode. TSAA then updates the threat database with this new information. With the minimum update rate of 1 Hz required by the standard, TSAA would be called in this mode at least once per second.
2. **Detect Mode:** TSAA is called in this mode when the currently tracked targets are to be evaluated as potential threats to the own-ship. TSAA can be called in

[13] As discussed later, a proposal is currently under evaluation by the DO-317 committee that would add constant turn rate extrapolation to the next version of the standard (DO-317B).

this mode independent of whether new target information is available – TSAA will use the most recent data available in the threat database.

Both modes of the TSAA algorithm are described in detail here from a functional and mathematical perspective.

Throughout the description of the algorithm, parameters that define the internal behavior of the algorithm are introduced and defined as variables (identified in italics). Depending on how those parameters are set, the algorithm will exhibit different alerting behavior. These parameters must be tuned and selected correctly to achieve the desired alerting behavior.

4.3.1 TSAA IN UPDATE MODE

When called in the 'Update' mode, TSAA updates the Threat Database with the state received state information. Even though the DO-317A tracker maintains a track for all active aircraft, a separate TSAA threat aircraft database is necessary for two reasons. First, it allows TSAA to maintain specific data locally within the TSAA alerting algorithm. Second, it allows TSAA to potentially pre-select which targets are maintained within it. For example, for computational reasons, a manufacturer may elect to only maintain targets that are within a predefined distance of the own-ship. The data fields maintained in the TSAA threat aircraft database are shown in Table 4-2. When TSAA is called in the update mode, the reports from the DO-317A tracker containing the data fields listed in Table 4-1 are used to fill the data fields for each of the active targets.

Table 4-2: Data Fields Maintained in the TSAA Threat Database

Data Field in TSAA Threat Database	Variable	Notes
Target ID	-	Call Sign, ICAO 24-bit address, Track ID from DO-317A tracker, or locally assigned ID
Time of Last Update	-	Time TSAA was lase called the 'Update' Mode
Time of Last ADS-B/ADS-R/TIS-B Message Reception	-	Last time new state information was received via ADS-B, ADS-R or TIS-B
X-Position	x	With reference to the local coordinate system in the DO-317A tracker
Y-Position	y	
Z-Position	z	
X-Velocity	$xdot$	
Y-Velocity	$ydot$	
Aircraft Ground Track	ψ	Angle between north and east velocity vectors

TSAA Estimated Turn Rate	$\dot{\psi}$	Turn rate as estimated by the TSAA turn rate estimator, estimated when TSAA is called in 'Detect' mode
TSAA Estimated Vertical Rate	v_z	Vertical rate as estimated by the TSAA vertical rate estimator, estimated when TSAA is called in 'Detect' mode

In addition to updating data for all existing targets in the database, TSAA also adds new targets that are not currently tracked or removes stale targets for which the data has become too old to be used for reliable conflict alerting. TSAA uses a maximum data age limit to determine when a target has become stale: if the difference between the time of last information update and the time of last ADS-B message reception is greater than that threshold, the target is discontinued. The threshold is denoted by the variable *TarDiscont*.

In the case of the exemplar algorithm, the database is implemented as a MATLAB structure that contains one object (an instance of the TSAA aircraft class defined in the tsaa_aircraft_class.m file) for each active target. The data fields listed in Table 4-2 in part define the object's properties.

4.3.2 TSAA IN DETECT MODE

When called in the 'Detect' mode, TSAA evaluates each target maintained in the TSAA threat database to determine whether it poses a threat to the own-ship. Adjusting the *ConflictSearchFreq* variable in the exemplar algorithm sets the frequency with which TSAA is called in this mode. Figure 4-6, which is itself an enlarged version of the Conflict Detector box in Figure 4-5, identifies the algorithmic components that are necessary to perform this evaluation.

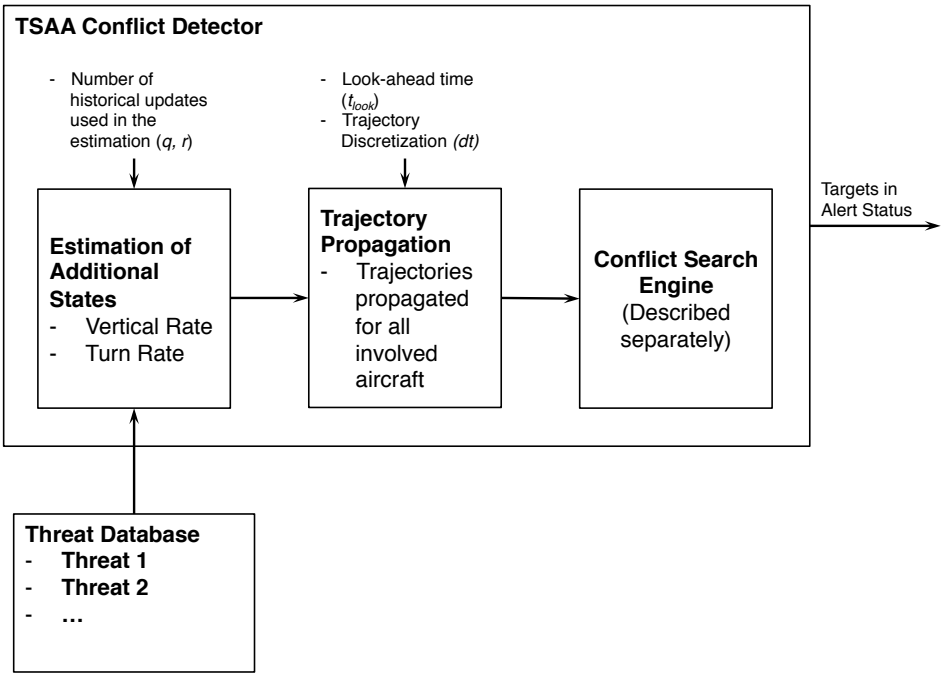


Figure 4-6: Functional Block Diagram of TSAA Conflict Detector

Each one of the components shown in Figure 4-6 is described in more detail in its own section below.

ESTIMATION OF ADDITIONAL STATES

As discussed in section 4.2, TSAA was required to estimate turn and vertical rate for each involved aircraft. In the exemplar algorithm, the following inputs and outputs are specific to this sub-routine:

Inputs:

- Historical Track for the own-ship and all targets
- Number of historical ground track angle and altitude values used in turn and vertical rate estimation (q, r)

Outputs:

- Estimated Turn Rate ($\dot{\psi}$)
- Estimated Vertical Rate (v_z)

The turn rate is calculated based on the differences in track angles between consecutive updates received by TSAA. To smooth out noise, a moving window filter using the last q

track angle differences is used to calculate the turn rate. The vertical velocity (v_z) is calculated with a linear regression fit over the *r* most recent altitude reports.

TRAJECTORY PROPAGATION

The trajectory generator creates the own-ship and target trajectories that define the path that the target or the own-ship would follow if it were to maintain the current turn rate and horizontal and vertical velocities.

Input:

- Historical Track for the own-ship and all targets
- Estimated turn and vertical rate
- Look-ahead time (t_{look})
- Trajectory Discretization (dt)

The trajectory propagation routine creates a trajectory consisting of discrete points in space, dt seconds apart, t_{look} seconds into the future. The number of points (*nPoints*) in the projected trajectory depends on look-ahead time and the trajectory discretization (dt) and is calculated by dividing t_{look} by dt. The first point of each trajectory is at the position last received by TSAA (x_0, y_0, z_0). For each trajectory, the following parameters are calculated and stored for later use:

Output:

- Time vector for the span of the trajectory (contains one entry for each point along the trajectory)
- For each time step in the time vector:
  - x, y, z coordinates of the trajectory
  - North, East and vertical velocities

The ground track angle (ψ), *x, y* and *z* positions of the trajectories for the own-ship and target are calculated for the length of the trajectory using Equation 1. The subscript *k* refers to the time step along the trajectory. The velocity *v* is the magnitude of the horizontal velocity and v_z is the vertical velocity. As mentioned, v_z is assumed to remain constant along the predicted trajectory.

Equation 1: Formulas to Calculate the Ground Track Angle (ψ), x, y and z Positions of the Trajectory

$$\left.\begin{aligned} \psi_k &= \psi_{k-1} + \dot{\psi} \cdot dt \\ x_k &= x_{k-1} + v \cdot \cos(\psi_{k-1}) \cdot dt \\ y_k &= y_{k-1} + v \cdot \sin(\psi_{k-1}) \cdot dt \\ z_k &= z_{k-1} + v_z \cdot dt \end{aligned}\right\} \quad for\ k = 1..nPoints$$

Once the trajectories are generated they are passed to the conflict search engine.

4.3.3 CONFLICT SEARCH ENGINE

The conflict search engine conducts pairwise comparisons between the own-ship trajectory and each target trajectory in order to determine whether any of the targets pose a threat to the own-ship. The conflict search engine contains the following subroutines, also described in Figure 4-7:

1. Calculation of PAZ size and physical separation
2. Evaluation of predicted separation against Collision Airspace Zone (CAZ) and Protected Airspace Zone (PAZ)
3. Determination of alert status of target

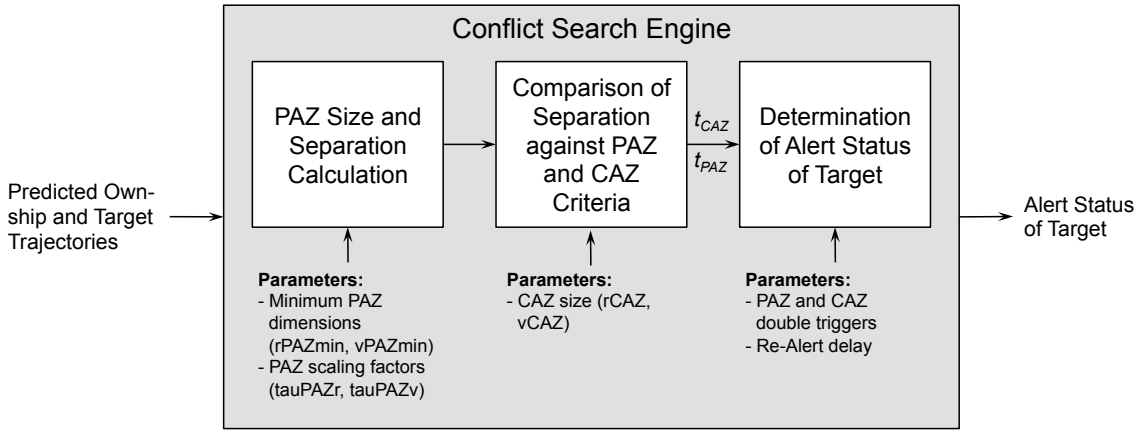


Figure 4-7: Functional Block Diagram of TSAA Conflict Search Engine

Each one of the routines is described in detail below.

CALCULATION OF PAZ SIZE AND PHYSICAL SEPARATION

Using the trajectories generated by the Trajectory Generator, this routine calculates the size of the PAZ around the target for each time step in the trajectory as well as the physical

separation between the own-ship and target. To calculate the size of the PAZ, the following input parameters are required:

Inputs:

- Minimum horizontal PAZ dimension (*rPAZmin*)
- Minimum vertical PAZ dimension (*vPAZmin*)
- Horizontal PAZ scaling factor (*tauPAZr*)
- Vertical PAZ scaling factor (*tauPAZv*)

For each time step along the trajectories, the following values are calculated:

Outputs:

- Horizontal size of the PAZ (*rPAZ*)
- Vertical size of the PAZ (*vPAZ*)
- Predicted Horizontal separation between the two aircraft (*hSep*)
- Predicted Vertical separation between the two aircraft (*vSep*)

Equation 2 provides the formula by which the horizontal closure rate is calculated. *d* denotes the separation distance in the respective dimension and *v* denotes the magnitude of velocity. The closure rate has to be calculated for each time step *i* along the trajectory because it can change due to the constant turn rate prediction for each trajectory. Additionally, the relative vertical velocity (*crv*) is calculated as the difference in the vertical rates of the two aircraft. Note that the relative vertical position matters when calculating the vertical closure rate.

Equation 2: Formula for the calculation of the horizontal closure used by TSAA

$$d_j = j_{target} - j_{own-ship} \qquad where \quad j = x, y$$

$$v_{rel,x} = v_{E,target} - v_{E,own-ship}$$

$$v_{rel,y} = v_{N,target} - v_{N,own-ship}$$

$$crh_i = \max\left(0, \frac{d_{x,i} \cdot v_{rel,x,i} + d_{y,i} \cdot v_{rel,y,i}}{\sqrt{d_{x,i}^2 + d_{y,i}^2}}\right)$$

Once the closure rate is calculated, the size of the PAZ along the trajectory is calculated using Equation 3. Figure 4-8 shows an example of how the size of the PAZ can change along the predicted trajectory. The solid red and blue lines are the historical tracks of the target

and own-ship, respectively. The black dotted line is the predicted trajectory of respective aircraft.

Equation 3: Formula to calculate horizontal and vertical PAZ size

$$rPAZ_i = rPAZmin + tauPAZr \cdot crh_i$$
$$vPAZ_i = vPAZmin + tauPAZv \cdot crv$$

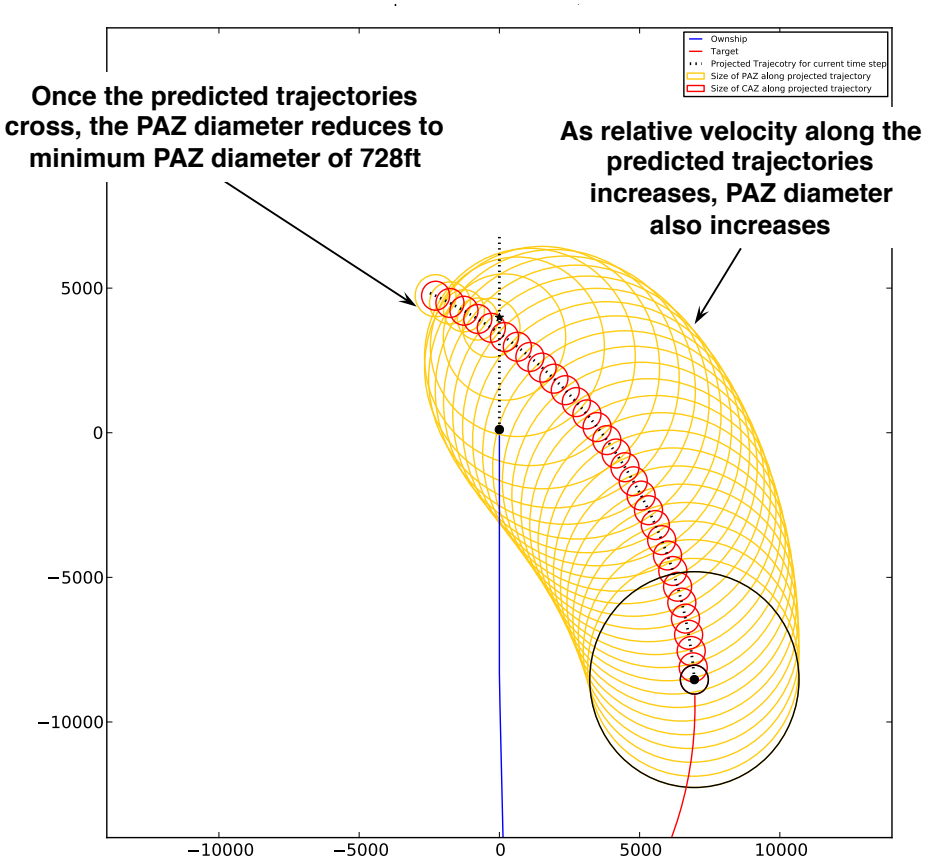


Figure 4-8: Visualization of PAZ Size for a Sample Encounter

To calculate the predicted vertical and horizontal separation along the trajectories the formulas in Equation 4 are used. Here again, *d* denotes distance and *z* denotes altitude.

Equation 4: Calculation of horizontal and vertical separation along the trajectories

$$hSep_i = \sqrt{d_{x,i}^2 + d_{y,i}^2}$$

$$vSep_i = abs\left(z_{own-ship|i} - z_{target|i}\right)$$

EVALUATION AGAINST CAZ AND PAZ VIOLATION CRITERIA

This routine compares the predicted separation distances (*hSep* and *vSep*) for each step along the trajectory to determine whether or not they lie within the bounds defined by the PAZ or the CAZ along the length of the trajectories. If both *hSep* and *vSep* are less than *rPAZ* and *vPAZ* (i.e., there is a PAZ violation), the earliest time along the trajectory that this is predicted is recorded as t_{PAZ}. Similarly, *hSep* and *vSep* are evaluated against *rCAZ* and *vCAZ*, and if a CAZ violation is predicted, the earliest time along the trajectory that it is predicted is recorded as t_{CAZ}.

DETERMINATION OF ALERT STATUS OF TARGET

If a violation of the PAZ or the CAZ is predicted, the values of t_{CAZ} and t_{PAZ} are evaluated against the alert logic described in this section. The alert logic determines whether to issue an alert for a given target and has the following main components:

1. Determine if an alert is necessary due to two consecutive predictions of either the PAZ or the CAZ
2. Determine if an immediate PAZ or CAZ alert is necessary
3. Determine if the alert occurred within the alert hysteresis and should thus be suppressed

In order to evaluate t_{CAZ} and t_{PAZ} against the three conditions above, the following inputs are necessary:

Inputs:

- Threshold value before which two consecutive PAZ or CAZ violations need to be predicted (*doubleTrigger*)
- Duration of time after an alert during which a second alert cannot be issued for the same target (*reAlertDelay*)

The output of this final routine is whether or not a target is in PAZ or CAZ alert status.

DETERMINE IF AN ALERT IS NECESSARY DUE TO TWO CONSECUTIVE PREDICTIONS OF VIOLATIONS OF EITHER THE PAZ OR THE CAZ

To reduce nuisance alerts, a "double trigger" value is introduced. If t_{CAZ} and t_{PAZ} are in excess of their double trigger thresholds, a violation of the PAZ or the CAZ has to be predicted by two consecutive conflict searches in order for the target to change into an alert state. In other words, the algorithm has to be called twice in a row and predict that the target will violate the PAZ or the CAZ both times for that target to be switched into alert state.

DETERMINE IF AN IMMEDIATE PAZ OR CAZ ALERT IS NECESSARY

For an alert to be issued on a target upon the first prediction of a violation (i.e., the algorithm runs only once), t_{CAZ} and t_{PAZ} have to be below the value of the double trigger threshold. For example, if the PAZ double trigger is set at 15 seconds, a PAZ conflict has to be predicted less than 15 seconds into the future (i.e., $t_{PAZ} < 15$) for the target to be switched into alert state upon the first prediction of a violation.

DETERMINE IF THE ALERT OCCURRED WITHIN THE ALERT HYSTERESIS AND SHOULD THUS BE SUPPRESSED

Once a target is identified as being in conflict with the own-ship and an alert is issued, the audio system announcing the alert is busy for the length of the annunciation of the alert. If a maneuver causes the target to go in and out of alert status during that time, a second alert is issued on the same target, queued, and then announced once the annunciation of the first alert is finished. Though this would be a legitimate alert, it is unnecessary and therefore not desired.

In order to prevent this queuing of alerts, a hysteresis is introduced and set such that once a target switches into alert state it will remain in alert state for the longest possible duration of the aural message. The hysteresis also prevents the target from changing appearance visually during the duration of the annunciation of the aural alert in class II systems.

The duration of the hysteresis is set by the value of the *reAlertDelay*. If $t_{now} - t_{last\ alert} <$ *reAlertDelay*, the alert status of the target is retained – even if the basic algorithm no longer predicts a conflict for that target.

4.4 Summary of Internal Algorithm Parameters

The algorithm described above has a total of 14 internal parameters that define the exact behavior of the algorithm. A summary of all parameters is provided in Table 4-3. If selected correctly, those parameters will allow the algorithm to operate at its best performance and provide the desired alerting performance defined in the system requirements.

Table 4-3: Algorithm Internal Parameters That Define Algorithm Behavior

Algorithm Parameter	Variable	Purpose
Look-ahead Time (s)	*tlook*	Length of predicted trajectory
Trajectory Discretization (s)	*dt*	How frequently discrete points are generated along the trajectory
Turn Rate Filter (#)	*q*	How many historical ground track angles are used for turn rate calculation
Vertical Rate Filter (#)	*r*	How many historical altitudes are used for vertical rate calculation
CAZ Radius (ft)	*rCAZ*	Radius of inner airspace buffer zone
CAZ Height (ft)	*vCAZ*	Height of inner airspace buffer zone
Min. PAZ Radius (ft)	*rPAZmin*	Minimum radius of outer airspace buffer zone
Min. PAZ Height (ft)	*vPAZmin*	Minimum height of outer airspace buffer zone
Hor. PAZ Scaling (s)	*tauPAZr*	How many seconds are used to scale the outer airspace buffer zone with closure rate horizontally
Vert. PAZ Scaling (s)	*tauPAZv*	How many seconds are used to scale the outer airspace buffer zone with closure rate vertically
Double Trigger (s)	*doubleTrigger*	If a conflict is predicted further into the future than this threshold, at least two consecutive predictions of PAZ or CAZ violation are required
Re-alert Delay (s)	*reAlertDelay*	Minimum duration of an alert
Target Discontinuation Threshold (s)	*TarDiscont*	Maximum time after last ADS-B message reception when a target is considered active
Conflict Search Frequency (s)	*ConflictSearchFreq*	Frequency at which TSAA is called

As discussed in the next chapter, it is not clear a priori how those parameters should be set and how they individually affect the various performance characteristics defined in the previous chapter. Therefore, a method to analyze and evaluate the parameter space defined by the parameters as well as the resulting performance space will be developed in the next chapter. However, some of the parameters do not necessarily depend on how they trade with other parameters or performance requirements, but rather are set by external elements and influences. Those parameters are summarized in the next section.

4.4.1 HARD-CODED PARAMETER SETTINGS

Table 4-4 lists the parameters that were not subjected to the tuning procedure described in the next chapters; the settings for these parameters were selected as part of the development process.

Table 4-4: Algorithm Parameters with Hard-Coded Settings

Algorithm Parameter	Parameter Setting
Conflict Search Frequency (s)	Once per second
Target Discontinuation Threshold (s)	15 sec
CAZ Radius (ft)	500 ft
CAZ Height (ft)	±200 ft
Re-alert Delay (s)	6 sec
Double Trigger	15 sec

The setting of the conflict search frequency was influenced by the fact that TSAA will receive updated state information at least once per second from the DO-317A tracker. Since TSAA should be run whenever new information becomes available, a corresponding frequency of once per second was selected for calling TSAA.

The Target Discontinuation Threshold was selected based on the fact that new information becomes available once every 12 seconds for a TIS-B target that is under surveillance by a single en-route radar. In order to allow alerting on such a target, the target discontinuation threshold was set at 15 seconds, allowing a buffer of 3 seconds in case the TIS-B ground system requires additional time to uplink the radar data. In the case that a single update from the radar is missed, this target would be discontinued.

As described at the very beginning of this chapter, the CAZ height and radius were set by maximum allowable horizontal position uncertainty of two rule compliant ADS-B targets (NACp of 8) and the vertical quantization of altitude to 100ft for most General Aviation aircraft.

The re-alert delay is set by the maximum possible length of the aural message that would have to be pronounced to the flight crew in response to an alert. During the duration of that aural annunciation, the traffic should remain in alert state to prevent the queuing of alerts and the turning off of the yellow, visual indicator in Class II system.

Lastly, the double trigger value was set to achieve the desired 12.5 seconds time of alert before closest point of approach for alerting systems. If a conflict is predicted less than 15

seconds into the future, TSAA alerts the first time it predicts an alert but if the conflict is predicted more than 15 seconds into the future, a second prediction is required.

Chapter 5

DEVELOPMENT OF ALGORITHM TUNING AND PERFORMANCE EVALUATION METHOD

The performance of the exemplar TSAA algorithm presented in the previous chapter can be adjusted by changing the internal algorithm parameters shown in Table 5-1. However, it is not clear, a priori, to what value those parameters should be set and how each parameter individually affects the various performance characteristics defined in section 3.4. Therefore, the TSAA design process included developing a method to evaluate those trade-offs, which this chapter will describe. The next chapter will then apply this method to the TSAA exemplar algorithm.

Table 5-1: Adjustable Parameter Internal to the Prototype TSAA Algorithm

	Algorithm Internal Parameter	Setting
Adjustable Parameters	Look-ahead Time (s)	Must be determined
	Trajectory Discretization (s)	
	Turn Rate Filter (#)	
	Vertical Rate Filter (#)	
	Min. PAZ Radius (ft)	
	Min. PAZ Height (ft)	
	Hor. PAZ Scaling (s)	
	Vert. PAZ Scaling (s)	
Preset Parameters	Conflict Search Frequency (s)	Once per second
	Target Discontinuation Threshold (s)	15 sec
	CAZ Radius (ft)	500ft
	CAZ Height (ft)	±200ft
	Re-alert Delay (s)	6 sec
	Double Trigger	15 sec

5.1 Trading Multiple Competing Performance Metrics

The process of optimizing a system's performance given a set of inputs and performance metrics has been studied extensively and is documented well in literature [93]. One very common approach to optimize a system with multiple performance metrics is multi-attribute utility theory (MAUT), in which a utility function that combines the performance metrics into a single value is defined and maximized. To generate such a function, assumptions must be made first about the relative utility of the various performance metrics. Given such relative weights, the performance metrics are then combined into a single function that allows the utility of a given system implementation to be calculated. A priori, no such utility function had been defined for TSAA.

Another approach currently under active development takes a more direct approach to identify the relationships between input and output variables, and is borrowed from methods in sensitivity analysis. Known as High Dimensional Model Representation (HDMR), it is "a set of quantitative model assessment and analysis tools for capturing high-dimensional input-output system behavior." [94] Given a set of input data, the HDMR approach allows for the identification of the input variables or parameters that contribute most significantly to the variance in the output. Based on the HDMR results, an in-depth analysis of the relationships between those high-impact inputs and how they trade with outputs can be conducted, and their setting tuned to achieve desired system behavior.

For the development of TSAA, the HDMR approach was used. A significant advantage of the HDMR approach over the MAUT method is that a key part of it is the direct visualization of the important parameters and performance trade-offs. This advantage makes it the more appropriate method to be used in the context of the consensus-based certification process used for TSAA.

5.2 General Set-Up of the Parameter Tuning Problem

The objective of the method is to define a way to tune the performance of a complex alerting system with multiple internal parameters and multiple competing performance attributes. The method takes a given system and adjusts its internal design parameters in the presence of changing state uncertainties and flight dynamics to find the parameter combination that best meets performance requirements. This is achieved by evaluating the n-dimensional trade space defined by the n system-internal parameters and mapping it to the m-dimensional performance space defined by m performance metrics. This method is intended to be a tool that enables an analyst to visualize and evaluate the trade-offs among

system parameters as well as the trade-offs between parameters and performance requirements for the alerting system of interest.

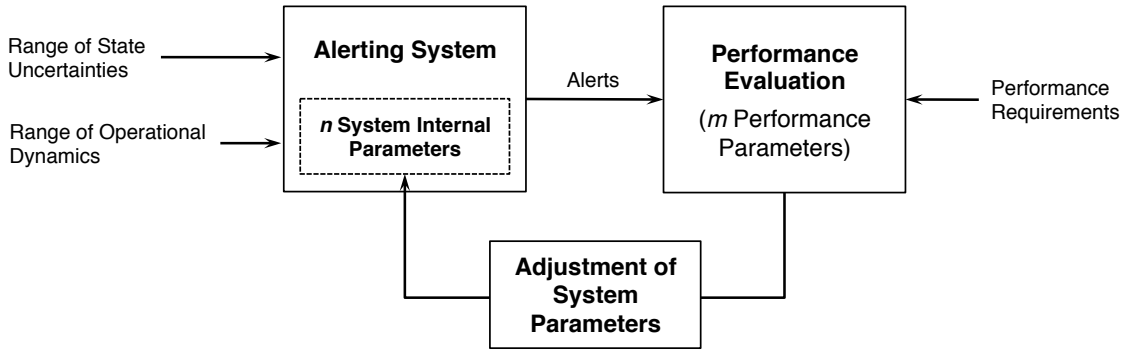


Figure 5-1: Schematic Representation of Optimization Approach Applied to a Conflict Alerting System

The system inputs to the TSAA algorithm are the state information of the own-ship and all the targets of interest. As discussed in Chapter 2, the level of current state uncertainty present in the information may be different depending on the sensor that was used to determine the state information. Additionally, the dynamics of the current operations that the aircraft of interest conducted are represented by the time-evolution of the state information. As shown in Figure 5-1, these state uncertainties and changing operational dynamics can be represented notionally as two separate inputs to the alerting system. For TSAA, the optimization variables are the 8 internal algorithm parameters that are still to be tuned (Table 5-1). Based on the issued alerts and the technical performance measures defined in Chapter 3**,** the system's performance then can be evaluated for different settings of the internal algorithm parameters. Figure 5-2 shows where the method development fits within the overall TSAA development process.

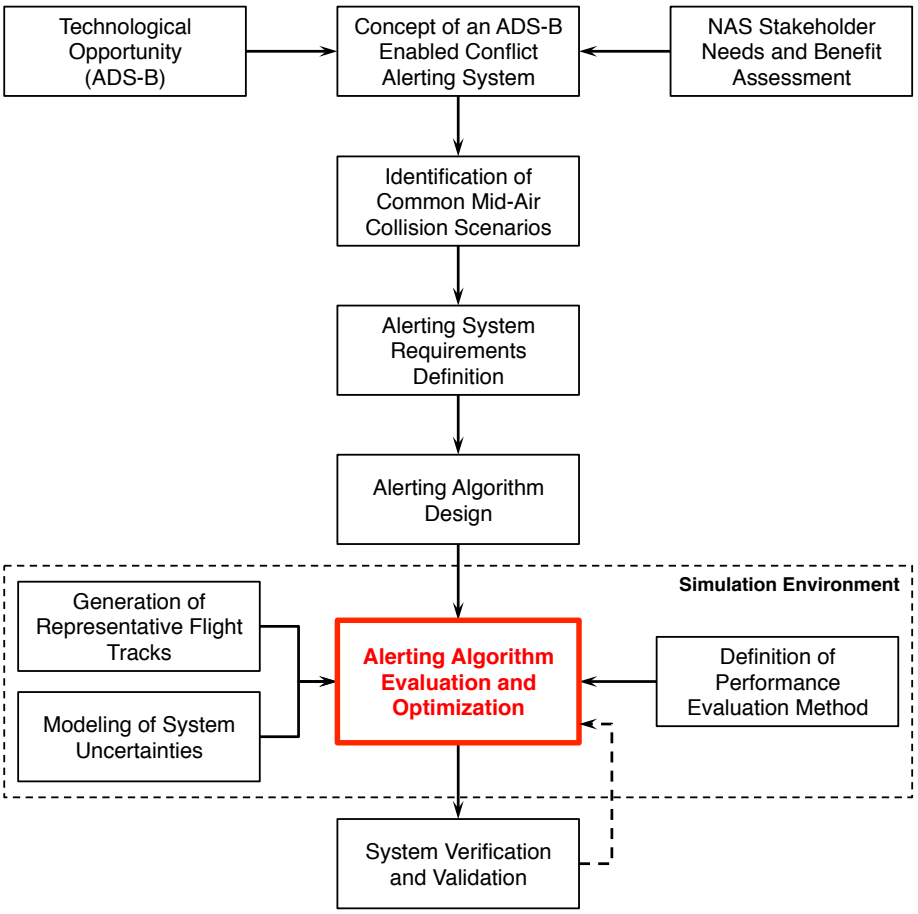


Figure 5-2: The Tuning Method is Used for the Evaluation and Optimization of the TSAA Algorithm

5.3 Conceptual Description of the TSAA Algorithm Tuning Method

Figure 5-3 shows the three major components of the algorithm tuning method. As described in section 5.2, the *n* internal system parameters define the *n*-dimensional parameter space. In the case of the TSAA exemplar algorithm, this space is an 8-dimensional hyperspace, representing the remaining 8 parameters to be tuned. For a particular point in the hyperspace, identified by an 8-dimensional vector, the behavior of the algorithm is modeled and the technical performance metrics are calculated by setting the algorithm parameters in the simulation environment to those values. In the *m*-dimensional performance space, the performance of a particular parameter combination can then be visualized. The process is

repeated for a different set of algorithm parameters, resulting in additional performance points in the performance space.

Based on the performance points in the performance space, in-depth analysis can then be performed to evaluate the relationships between the individual algorithm parameters and the alerting performance.

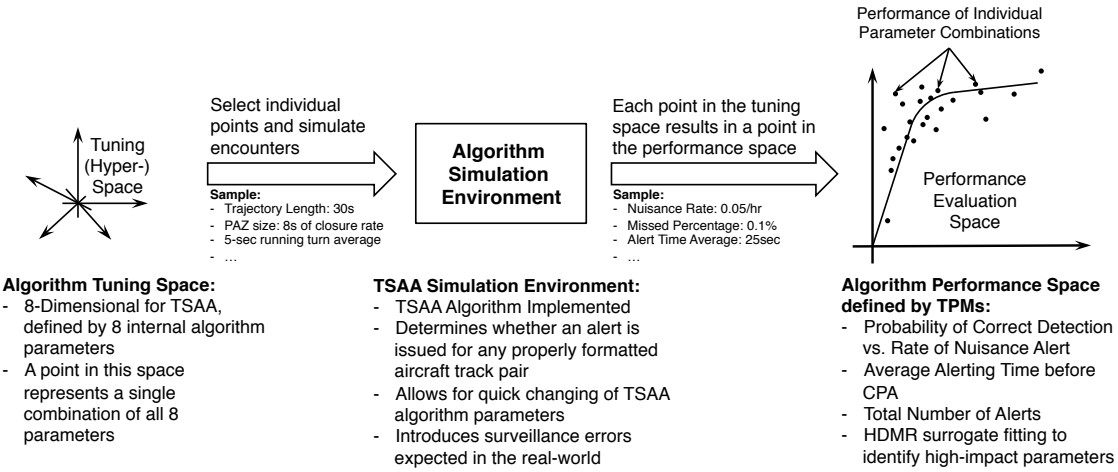


Figure 5-3: Algorithm Tuning Method

Each of the three steps shown in Figure 5-3 is described in detail in the following sections.

5.4 Step 1: Parameter Space Sampling Method

Defining a range over which a particular parameter can vary reduces the *n*-dimensional hyperspace to an *n*-dimensional hyper*cube* from within which the parameter combinations must be selected. Even though the definition of a constraint region limits the size of the tuning space, it is computationally prohibitive to perform a full factorial evaluation of the parameter hypercube, especially considering that the range of each parameter can be quantized into an infinite number of segments.

One approach to reducing the magnitude of the computational task is to fix all parameters at a nominal value and then vary one parameter at time. This approach effectively performs a linear sensitivity analysis on how the performance responds to perturbations in a single parameter. Using terminology more commonly employed to describe the design of experiments, this approach is sometimes referred to as a "factorial experiment" or "One-At-A-Time Sampling" [94], [95]. However, this approach does not capture higher order

interactions between input parameters that may be present in complex alerting systems with high dimensionality. By way of example, as Figure 5-4 shows, in TSAA, a second order interaction may be present between the length of the trajectory (*tlook*) and the rate at which the PAZ scales (*tauPAZr)*.

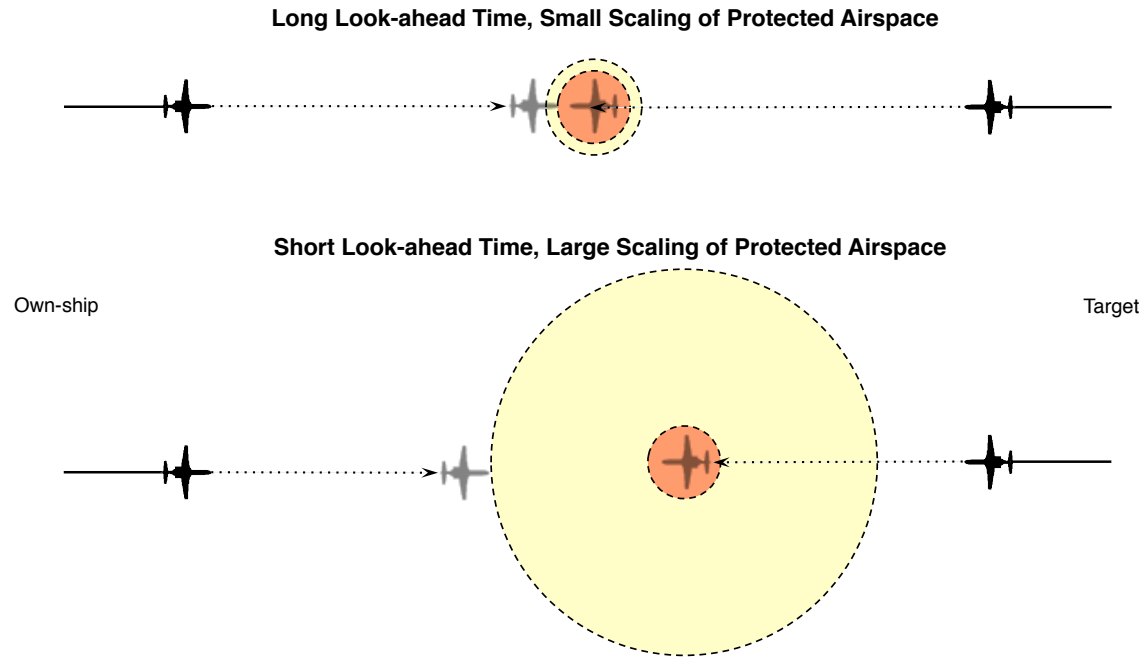


Figure 5-4: Sample Higher Order Parameter Interaction

Thus, the goal of the method to select the parameter combinations is to fill the entire space (i.e., hypercube) with as few samples as possible while still capturing all the effects of the parameters on system performance.

5.4.1 THE LATIN HYPERCUBE METHOD TO EFFICIENTLY SAMPLE THE TSAA PARAMETER HYPERCUBE

The Latin Hypercube method for efficiently sampling a large space was used as a means to address all the considerations listed above. Instead of randomly placing points throughout the space as a Monte Carlo method would, the Latin Hypercube method places one point at a time, minimizing overlap with previously placed points. As a result, each point provides a maximum amount of new information about a particular region in the parameter space [96].

Given a total number of desired points, the Latin Hypercube method generates a set of Hypercube points that span the entirety of the hypercube to be evaluated. In the case of

TSAA, each Hypercube point would be a combination of each of the 8 internal system parameters to be determined and by extension would be a different implementation of the algorithm. In the following sections, the terms *hypercube point*, *algorithm implementation*, and *parameter combination* will be used interchangeably.

The Latin Hypercube sample generator in MATLAB was used to generate the hypercube points for the analysis of the TSAA exemplar algorithm. After generating the initial sample, MATLAB offers the capability of refining the initial sample of hypercube points to minimize correlation between the points or to maximize the minimum distance between two points. For the sake of the TSAA analysis the points were refined by minimizing the correlation between parameters in an effort to prevent the effect of one parameter being overshadowed by another parameter changing at the same time in a correlated fashion.

Two factors influence the total number of hypercube points that are generated. First, the more points that are generated, the more simulations have to be run at a later time to evaluate all of those points. As later sections will discussed, each hypercube point requires the simulation of a full data set of airborne encounter to generate representative performance metrics, which is computationally intense. Second, if not enough points are generated, the parameter hyperspace may not be covered sufficiently, which results in insufficient data to evaluate the performance trade-offs later on.

In order to evaluate the 8 remaining parameters, the amount of 100 hypercube points was determined to be a reasonable balance between those two considerations and the most practical total from a computational perspective. During the initial phases of TSAA development, sets containing between 50 and 200 points were evaluated to determine whether the results changed significantly when different numbers of point were used. With 50 points, the performance space was populated very sparsely, which introduced difficulty in evaluating the trade-offs between performance metrics. When the number increased to above 100, the additional performance points remain in the region defined by the initial 100 points only providing a marginal benefit.

It should be noted that there are other methods to sample a hyper-dimensional space efficiently. The Latin hypercube lends itself well to our purposes due to its simplicity. For a deeper discussion on sampling methods, refer to references [95], [96].

5.5 Step 2: Suite of Tools for Algorithm Performance Simulation

Referring back to Figure 5-3, the next step in the algorithm tuning method is to determine the performance of a given parameter combination. To do so, a suite of tools for fast-time simulation consisting of the five components shown in Figure 5-5 was created.

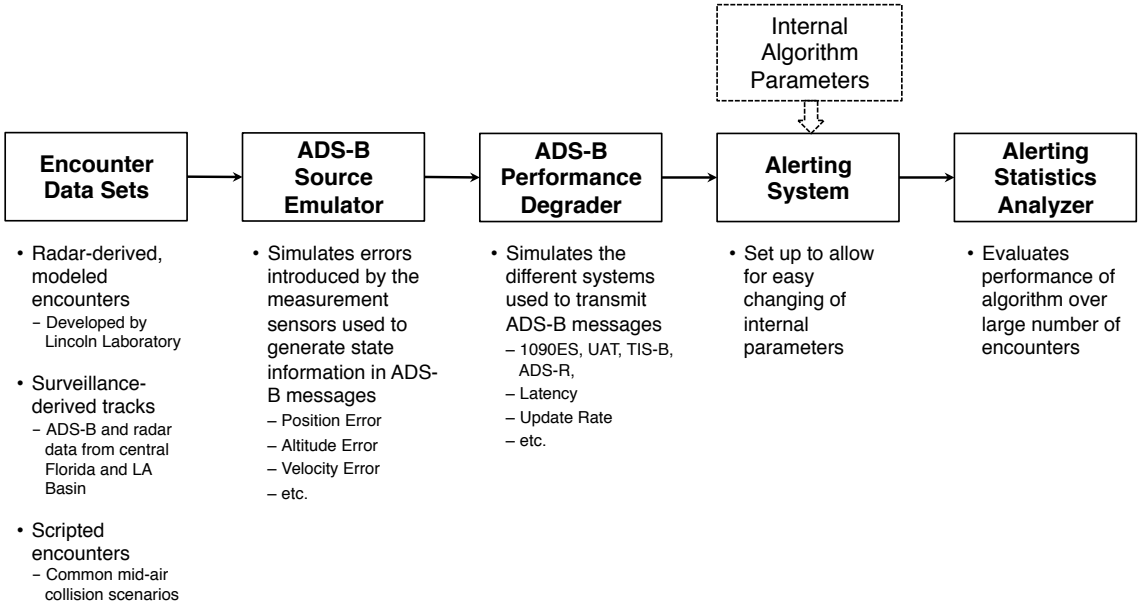


Figure 5-5: Simulation Tool Suite Used for TSAA Performance Evaluations

Flight tracks of airborne encounters are drawn from encounter data sets that were generated using encounter models, surveillance data, or the scenarios defined in Chapter 3. The flight tracks are fed through an ADS-B message source emulator and a performance degrader that introduce representative errors before the flight tracks are passed on to the alerting system. Last, the statistics analyzer evaluates the system's performance based on the alerts issued for the particular encounter data set. The following sections describe each component of the simulation tool individually.

5.5.1 ENCOUNTER DATA SETS

For an encounter data set to be representative, the encounters contained in it must accurately represent the encounters that the system is expected to experience when operating in the real world. As TSAA is expected to operate in all operational environments on-board a variety of aircraft, the data set should include operations from all environments and the operational characteristics and dynamics should be representative of the

performance of all of today's aircraft. A representative data set also allows for an accurate evaluation of an algorithm's ability to minimize predictive uncertainty. If the dynamics and operations are realistic, an algorithm that accurately predicts future states will perform better than an algorithm with poor ability to approximate future operations.

During the initial design of the algorithm, a basic data set consisting of modeled encounters was used to evaluate architectural and early design stage decisions. Once the algorithmic approach was solidified and only the parameter tuning remained, the basic data set was replaced with a data set consisting of actual radar tracks and known hazardous encounters. The early stage data set is referred to as the Lincoln Laboratory Encounter Model (LLEM) Master Encounter Set. The second data set is the Low Altitude and Airport Operations Data Master Encounter Set. Figure 5-6 shows the relationship between the tuning process and the encounter data sets.

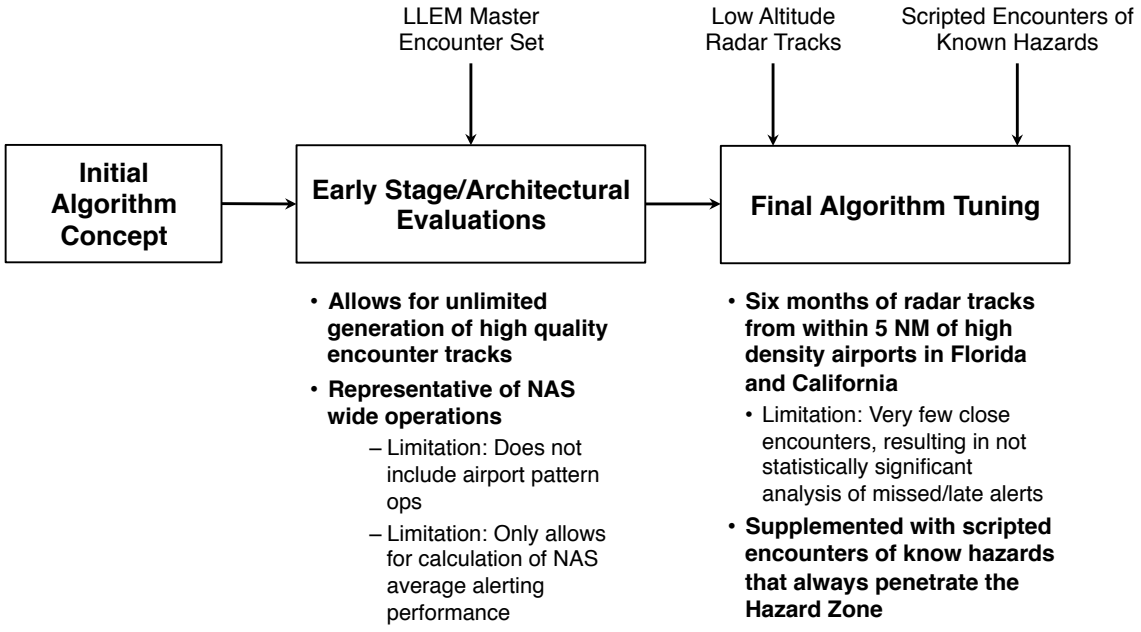


Figure 5-6: Relationship Between Encounter Data Sets and Algorithm Development Process

LLEM MASTER ENCOUNTER SET

Using 9 months of radar tracks from across the US, Lincoln Laboratory developed multiple encounter models that generate flight tracks that are statistically representative of encounters observed in the radar data[75], [76], [78]. The two models used to generate the

LLEM Master Encounter Set were the correlated encounter model (representative of aircraft operating while controlled by ATC) and the uncorrelated encounter model (representative of aircraft operating without ATC interaction). A total of 1,000,000 encounters were generated, 63% of which draw from the correlated encounter model and 37% from the uncorrelated. The percentages are representative of the frequency that correlated and uncorrelated encounters were observed during the 9 months of radar data. Figure 5-7 shows an example uncorrelated encounter from the LLEM.

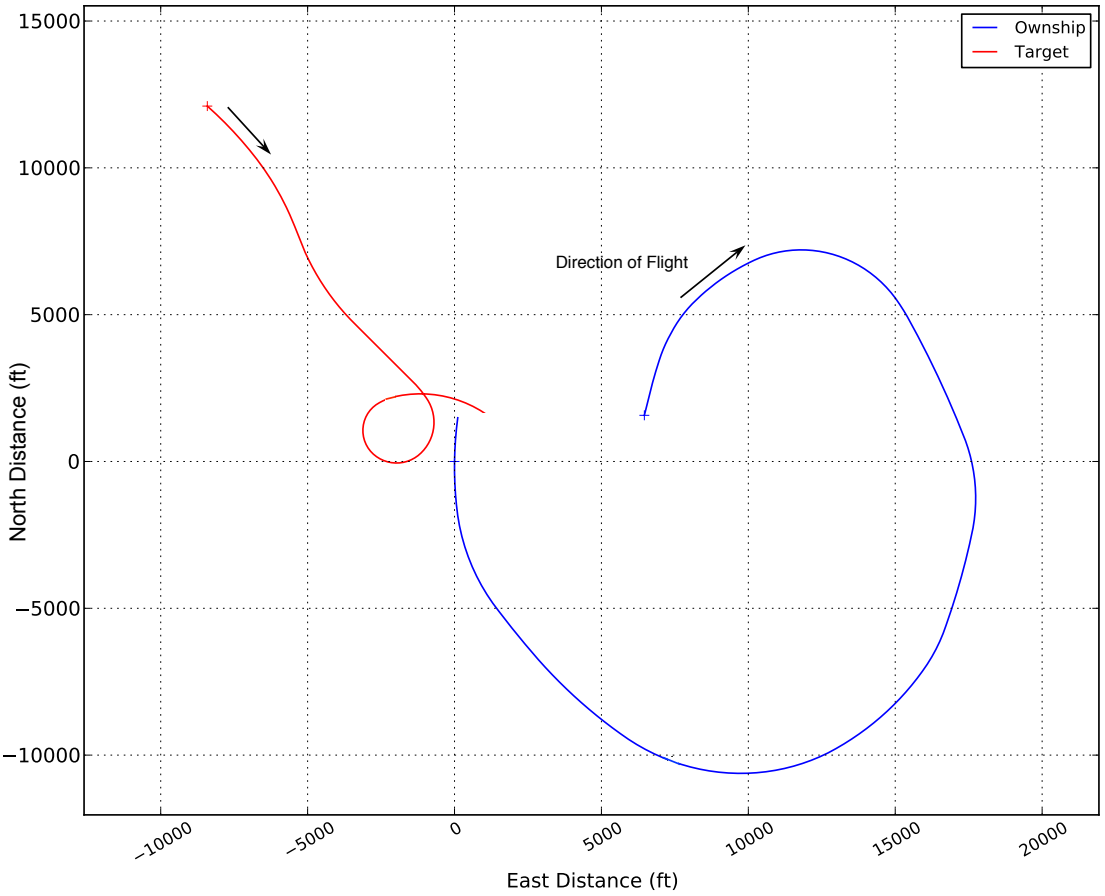


Figure 5-7: Sample Uncorrelated Encounter From the LLEM Master Encounter Set

All encounters in the LLEM Master Encounter Set contain 90 seconds worth of state data that describes two aircraft encountering each other. The state data is considered "truth data" since it does not contain any error.

The LLEM encounter models have two limitations. First, the models are not designed to generate encounters to represent operations in the airport pattern, which is the environment of most interest for TSAA.[14] Second, the data set is not well suited to evaluate rates of nuisance and overall alerts. Though an estimate for a NAS-wide average can be calculated, the rates are most frequently of interest in the high-density environments. During the initial development of TSAA, these limitations were recognized and once the in-depth analysis of the algorithm began, a data set that more accurately represents the environments of interest was created.

LOW ALTITUDE AND AIRPORT OPERATIONS MASTER ENCOUNTER SET

The Low Altitude and Airport Operations Master Encounter Set was generated using 6 months of surveillance data output by the fusion tracker of the SBS ground system in central Florida and the LA basin.[15] Most of the work to generate the data set was performed with Douglas Havens and David Elliott at the MITRE Corporation.

Six months of flight tracks were evaluated to identify those that operated within 5 nautical miles of an airport; each flight that did so was designated an "own-ship trip". The airports included controlled and uncontrolled airports. For each trip, the radar data was searched for all other aircraft that were operating within 20 NM of the own-ship during the duration of the trip. Those aircraft were included in the encounter as potential targets. All in all, this resulted in 300,000 own-ship trips during which the own-ship encounters between 1 and 182 other aircraft. Of the 300,000 trips, 83,000 are from Florida airports and 217,000 from California airports. The data set consisted of 44,195 own-ship flight hours. Figure 5-8 shows a sample own-ship trip with seven targets.

ADDITION OF SCRIPTED ENCOUNTERS TO IMPROVE POWER OF MISSED ALERT ANALYSIS

However, in a testament to the safety of operations in the NAS, certain types of encounters that are not common in normal operations (e.g., very close calls) are not well represented in the radar data. Were the data set to be used as is, this under-representation could result in an insufficient evaluation of the alerting system against this type of encounter. To address

[14] Though such encounters were present in the 9 months of radar data used to generate the models, they were excluded for two reasons: First, since they occurred more frequently, the models would have been skewed to generate more encounters in the pattern at the expense of encounter en-route. Second, the program under which these encounter models were generated was focused on the improvement of TCAS – since TCAS is suppressed at low altitudes, environments such as the airport pattern were not of primary interest.

[15] The data originated from the SBS surveillance volumes designated VCS2 and VCS11.

this concern, scripted encounters representing the known danger cases identified during the mid-air collision analysis were injected to the data set.

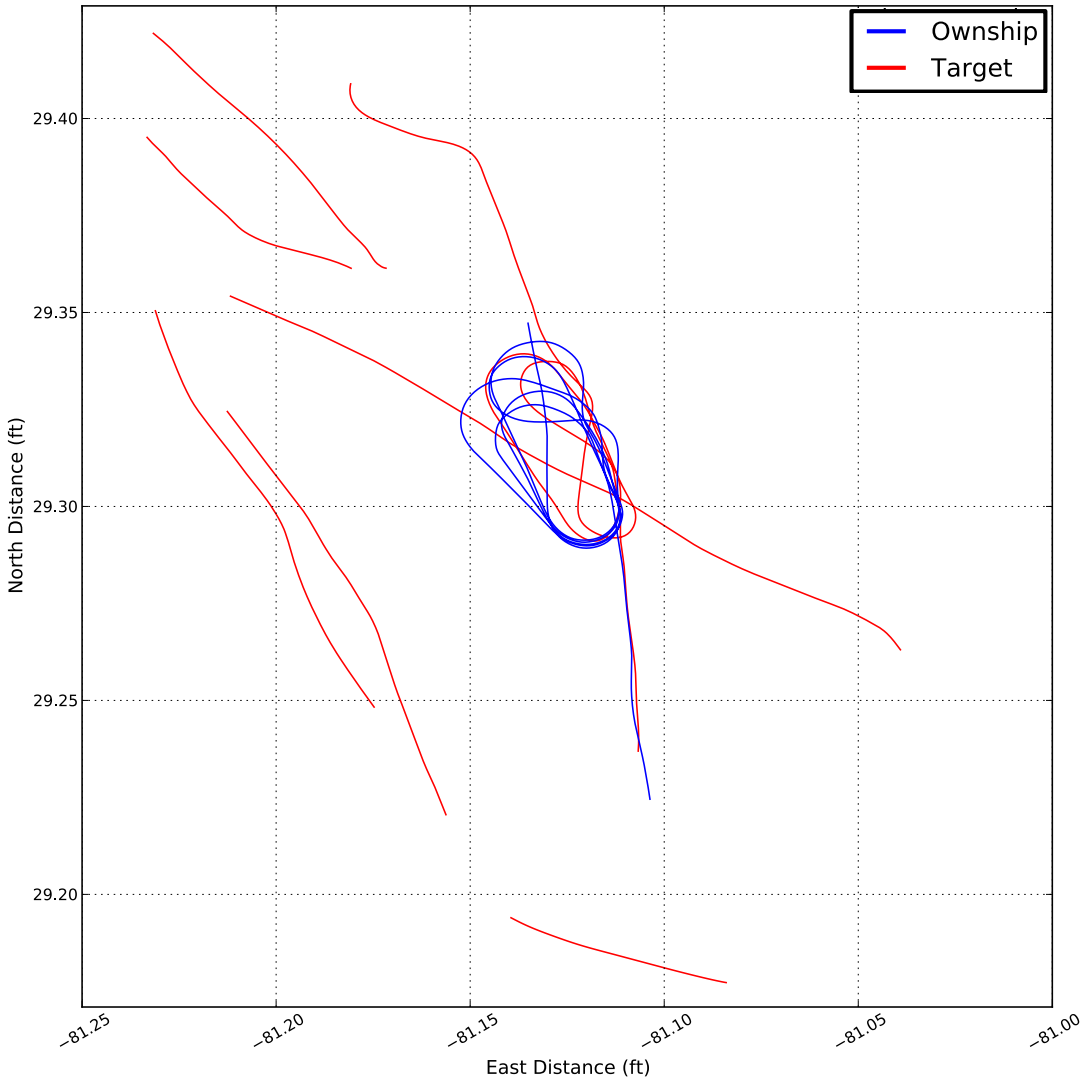


Figure 5-8: Sample Own-Ship Trip in the Low Altitude and Airport Operations Master Encounter Data Set

Of the 14 scenarios identified during the mid-air collision analysis, nine are specific to the airport environment. Using the parameters defined for the encounter geometries in Chapter 3, 10,000 encounters were generated for each of the nine encounters and added to the Low Altitude and Airport Operations Master Encounter Set. Figure 5-9 shows a sample encounter for encounter scenario A4.

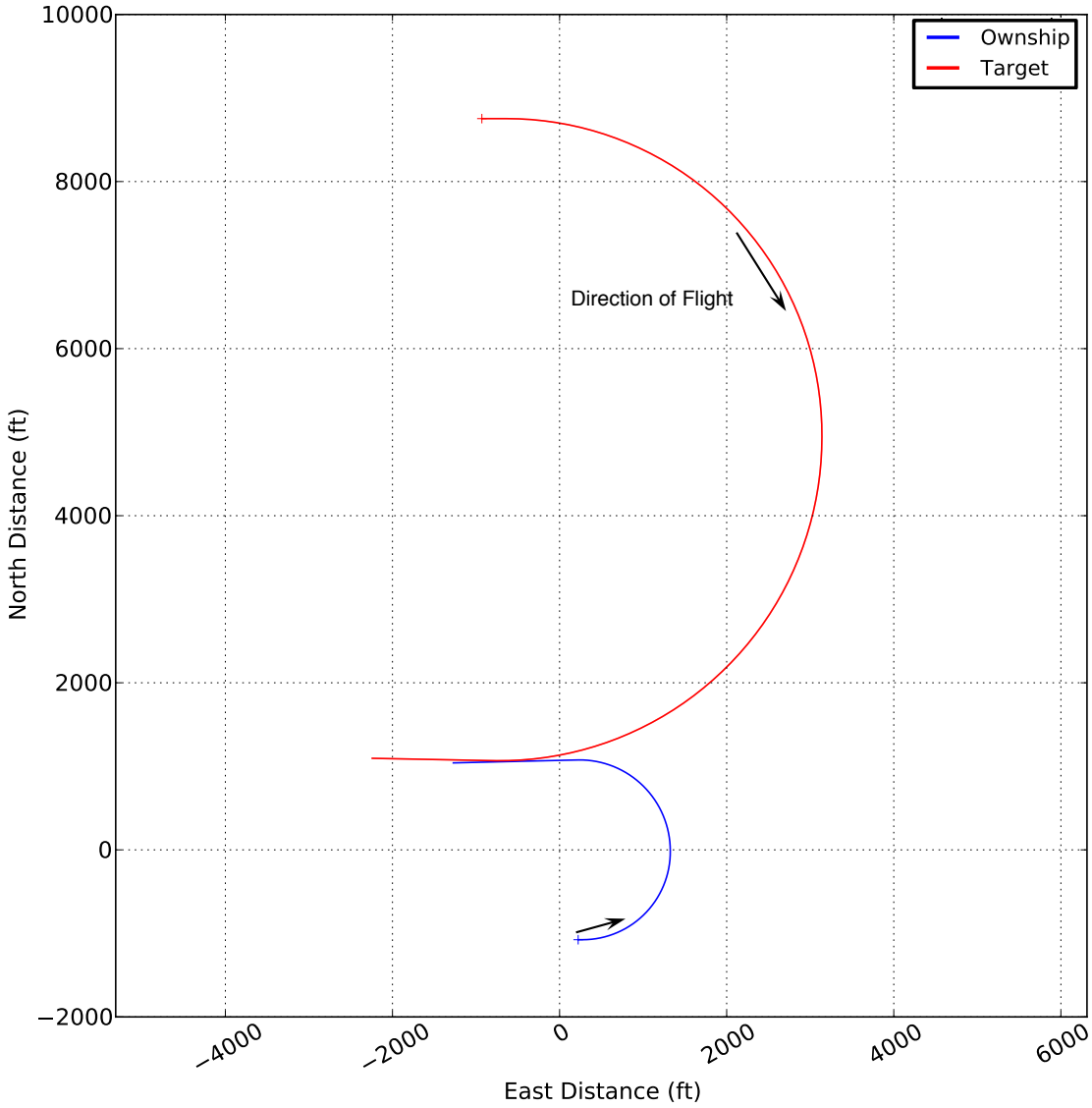


Figure 5-9: Sample Scripted Encounter of Scenario A4

Though introducing the additional encounters to the data set improves the statistical power of the missed alert analysis, it does introduce two limitations to the overall analysis. First, since the encounters were generated by hand, they do not contain representative flight technical variation as would be observed in a hand-flown flight track in the airport pattern. Introducing the error described in the next section reduces some of this effect. Second, the encounters do not contain any history as to how the two aircraft ended up in the encounter. For example, in the A1 scenario, aircraft operate in close proximity with low closure rates

for prolonged amounts of time. However, in order to end up in that type of geometry, the aircraft most likely will have closed in on each other at higher closure rates and different geometries beforehand, potentially generating an alert at an earlier time. Therefore, initializing the encounters in some of the scenario-defined geometries may not always truly represent the full encounter.

5.5.2 ADS-B SOURCE EMULATOR AND PERFORMANCE DEGRADER

Referring back to Figure 5-5, the flight tracks from the data set pass though the surveillance data source emulator and the ADS-B performance degrader. Together referred to as the "degrader," those two functions introduce representative errors to the state data in the encounter files. Figure 5-10 is a schematic representation of the degrader's process.

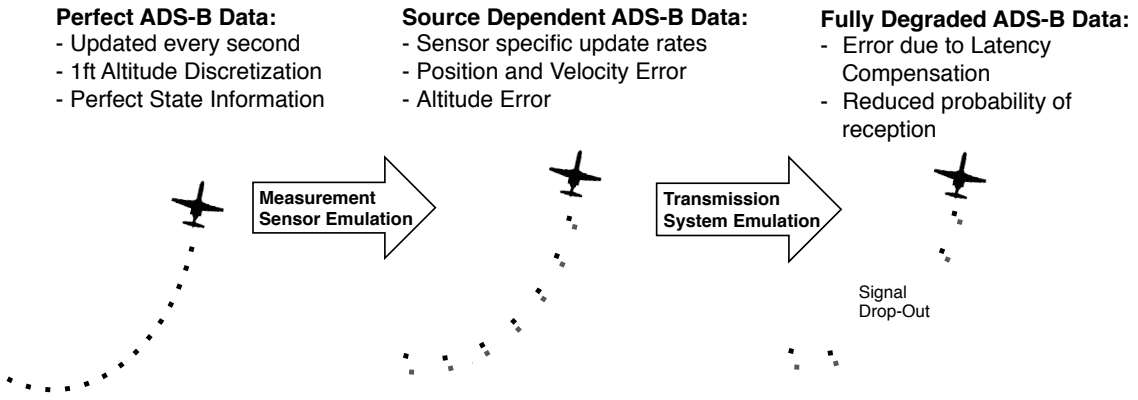


Figure 5-10: Schematic Representation of Degradation Process

Given the flight tracks from the encounter data set, the degrader first introduces state errors such as position, velocity, and altitude errors (notionally shown as the light gray "degraded" state data). Also, depending on the sensor from which the data originates, the frequency at which new data becomes available could be reduced. The second step introduces additional errors that result from the processing and transmission of the ADS-B data between the measurement sensor and TSAA.

The errors in the second step mainly result from the distributed nature of the larger ADS-B system architecture, which can affect ADS-B state data negatively. Figure 5-11 adapts Figure 2-3 to show a notional information flow for ADS-B data before it reaches the alerting system in an ADS-B–based alerting system. The current state uncertainty present in the data by the time it reaches the alerting system consists of the uncertainty in the original state measurement plus any additional uncertainty introduced by the data moving along this data flow.

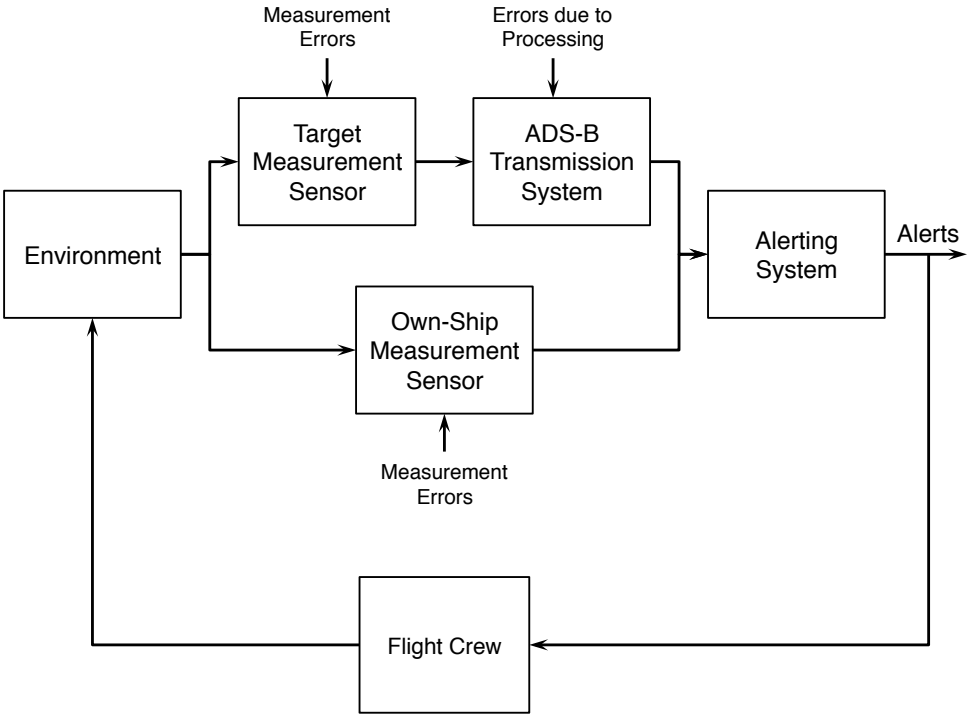


Figure 5-11: Schematic Representation of Functions and Information Flow of ADS-B Based Alerting Systems

Five different error models were added to the simulation tool to representatively model the state uncertainty present in the data by the time it reaches the alerting system in real-world operations. Listed below, three of the models are specific to the errors introduced during the initial state measurement and two are specific to the errors introduced due to the processing and transmission of the ADS-B data before it is received by the alerting system. The models were implemented such that they could simulate errors for radar or GNSS surveillance sources as well as for all three types of links available via ADS-B (aircraft to aircraft ADS-B, ADS-R and TIS-B). Each one of the models is described separately below.

1. Position Error (Measurement Error)
2. Velocity Error (Measurement Error)
3. Altitude Error (Measurement Error)
4. Error due to Latency Compensation (Processing Error)
5. Error due to Reduced Probability of Reception (Processing Error)

POSITION ERROR MODEL

The mechanism by which error is introduced to the position measurement and how that error behaves over time depends on the sensor taking the measurement. In the case of air-

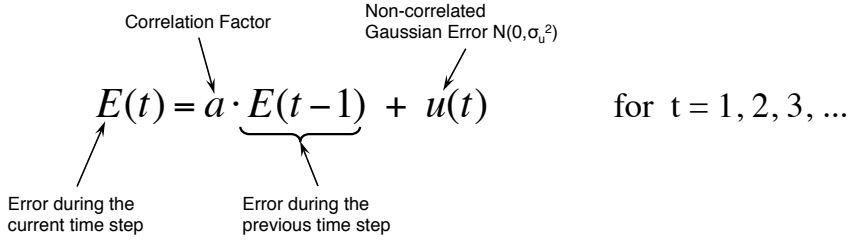
to-air ADS-B as well as ADS-R, the sensor used to determine the states is on-board the aircraft that originally transmitted the ADS-B message. In the case of TIS-B, the sensor is a ground based radar.

The ADS-B Out mandate does not specify which type of sensor must serve as the source of position and velocity information. As long as it meets the ADS-B performance requirements per the 2020 mandate, any sensor may be used. However, as is written in the mandate's introductory text [4]:

> *"... operators may equip with any position source. Although [GPS] WAAS is not required, at this time it is the only positioning service that provides the equivalent [reliability] to radar (99.9 percent reliability). The FAA expects that future position sources [...] will also provide 99.9 percent [reliability]."* [4]

In light of this, the position error model must be able to model position errors that are characteristic of GNSS (or "GPS") systems for ADS-B targets as well as those that are characteristic of ground based radar systems for TIS-B targets. As discussed earlier, ADS-B messages contain a measure of accuracy in terms of a NACp or a NACv, which allows the error coming from a GNSS sensor or radar to be modeled. Mohleji and Wang propose using a Gauss-Markov process to model the typically auto-correlated position measurement errors found in GNSS or radar systems [14]. Equation 5 shows the equation used to generate the position error at time *t* based on the error that was present at time *t-1*.

Equation 5: Error Model for Position and Velocity Errors as Proposed by Mohleji and Wang



E represents the magnitude of the error in a particular state (e.g., latitude or longitude, range or azimuth, etc.). As a result, in the horizontal plane, the measured position moves around the true position located at the origin in Figure 5-12, as the blue line's time evolution shows. *E* at time *t* is correlated to the error in the previous time step based on a correlation factor *a*. As shown in Figure 5-12 (left), as the correlation factor *a* decreases, the dependence on the previous error decreases as well, and the error becomes more Gaussian. On the contrary, a larger correlation factor results in a behavior that represents a time-

varying bias with jitter. The term *u(t)* behaves in a Gaussian fashion with a distribution of N(0, σ^2) where σ^2 adjusts each step in a manner to ensure *E* remains within the bounds specified by the NACp value. In general, GNSS position measurements are more auto-correlated (i.e., higher *a*) than radar position measurements.

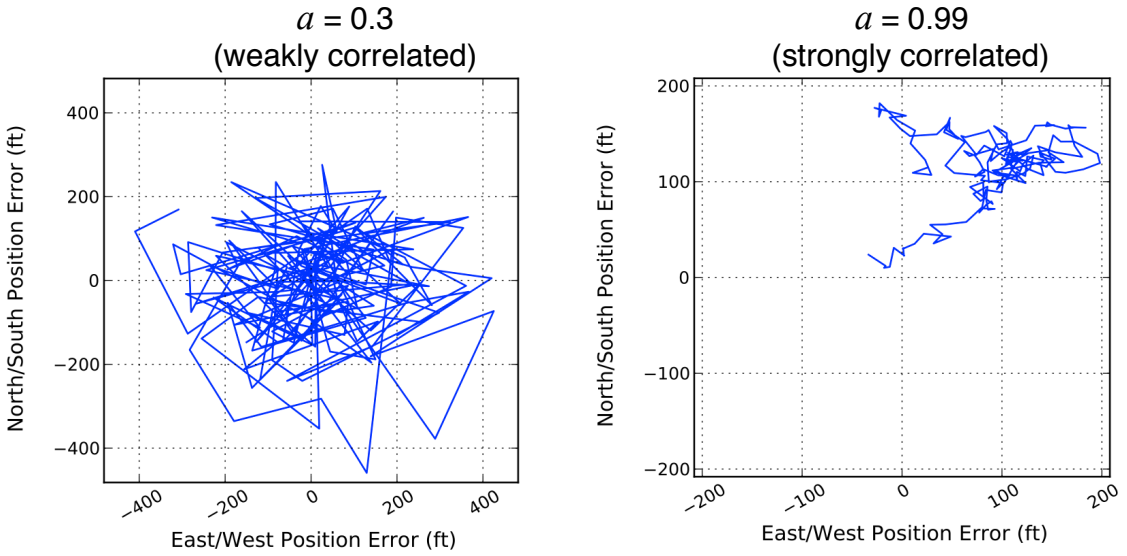


Figure 5-12: Weakly and Strongly Correlated Position Error, NACp of 8 (0.05NM)

As mentioned, the error model for the TSAA simulation environment must be capable of simulating errors from GNSS as well as ground based radar systems. Errors from radar systems are similar to those from GNSS systems in that they can be modeled as a slow moving bias with a jitter superimposed it. However, compared to GNSS systems, the correlation factor *a* is lower and the error thus behaves more Gaussian. In the model discussed here, the GNSS error is modeled with an *a* of 0.9966, which represents an auto-correlation time of 5 minutes, and the radar error is modeled with an *a* of 0.9780, representing an auto-correlation time of 45 seconds. These values were selected based on the Mohleji paper and input from radar experts at the MITRE Corporation. Figure 5-13 shows the time evolution of a radar error in red and a GNSS error in blue for a NACp of 8 and the auto-correlation time.

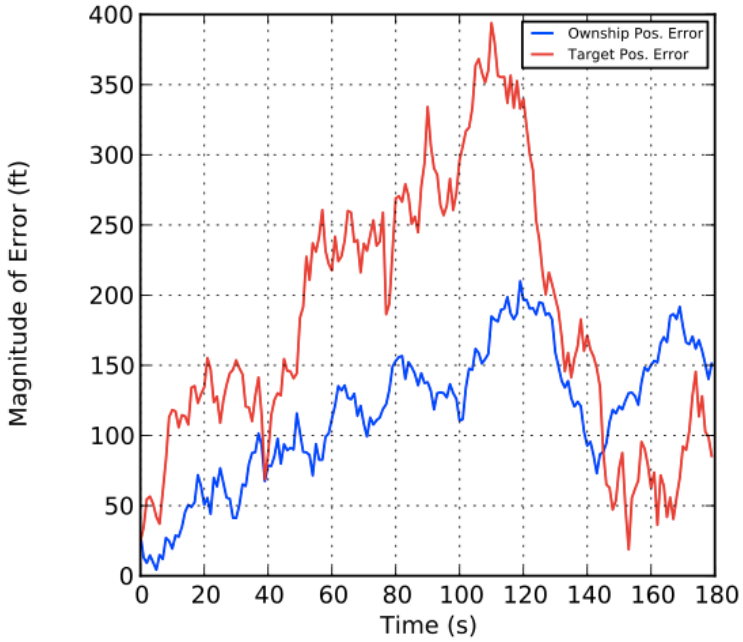


Figure 5-13: GNSS Error (blue) Compare to Radar Error (red) for NACp of 8

The geometry of the error distribution also depends on the sensor that initially generated the measurement. Radar systems measure the aircraft's range and azimuth in relation to the radar. Generally, the range measurement is more accurate than the azimuth measurement. As a result, the shape of radar errors tend to approximate an ellipse oriented perpendicular to the bore-sight of the radar. However, for GNSS sensors the error geometry may change as the satellite constellations change; as a result, their geometries can range from spherical to elliptical.

Figure 5-14 (left) shows a sample encounter with truth data (solid lines) and the degraded data (shaded lines). The own-ship error is modeled with a NACp of 8 and the target error is modeled with a NACp of 5.

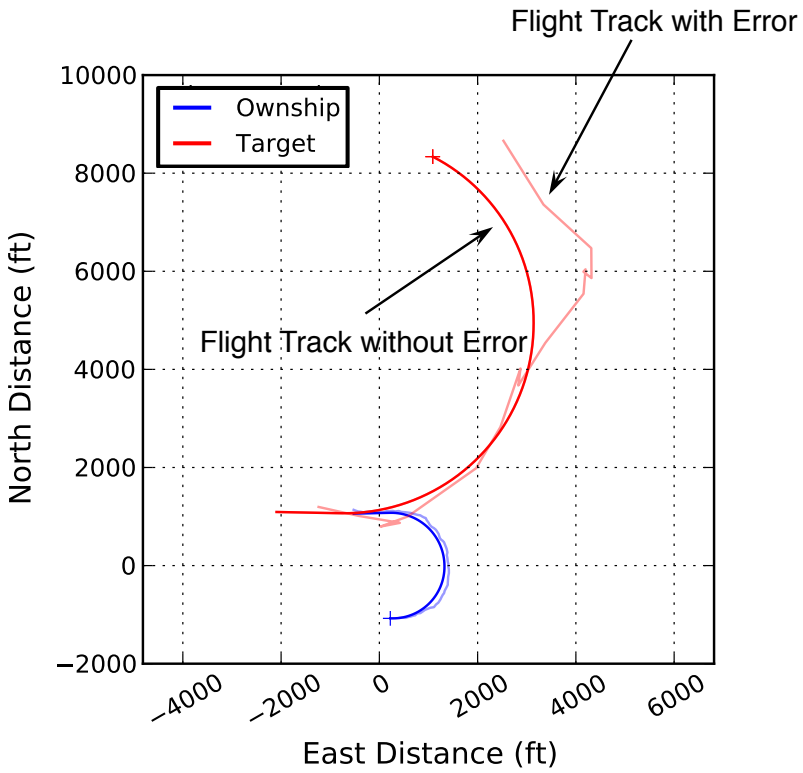


Figure 5-14: Sample Encounter With Position Error

VELOCITY ERROR MODEL

The errors in velocity derived by GNSS and radar systems are generally intimately related to the errors that are present in the position errors [97]. Therefore, similar to the position error, velocity errors can be represented using the Gauss-Markov model; with the only difference that instead of using the NACp value, the NACv value is used. As a reminder from Chapter 2, the reported NACv values can be as much as 4 times worse as the actual velocity errors for ADS-B targets using GPS systems. As such, modeling the velocity error based on reported NACv values is a conservative approach.

It is important to note that the position error affects TSAA in two main ways. First, the position error reduces the accuracy with which the current separation between the own-ship and the target can be determined. Second, it causes the predicted trajectories to originate at locations that do not represent where the aircraft truly are. However, the velocity error affects the predicted trajectories more directly; since the reported velocity is used to determine the direction of flight and thus the direction of the predicted trajectory, its errors can result in a predicted trajectory that does not align with the true direction of

flight. Additionally, errors in the magnitude of the velocity result in trajectories extrapolated for incorrect lengths, which could result in nuisance or missed alerts. As such, even though the velocity error is an error in the current state, it strongly affects the uncertainty in future states. Position and altitude errors more strongly affect the current state uncertainty and are then carried along but not necessarily magnified during the future state prediction.

ALTITUDE ERROR MODEL

A barometric sensor will measure the altitude transmitted via ADS-B. ADS-B includes a field containing geometric altitude but this altitude is only intended for verification purposes by ground systems [4]. The ICAO Annex 10 presents an error model describing altitude errors introduced by barometric sensor; this model has been adapted for the TSAA simulation tool [98]. Since altimetry errors are generally constant, the model samples an error from a Laplace distribution at the beginning of a simulation and then holds it constant during the simulated encounter. A sample own-ship and target Laplace distribution is shown in Figure 5-15.

ERROR DUE TO LATENCY COMPENSATION

The measurement error for position, velocity and altitude is present in each measurement when it is taken. However, due to the dynamic nature of flight operations, this information has a limited time of usefulness. The more time that passes between performing the measurement and using the information in that measurement, the less relevant the information becomes. Stated differently, as time elapses, the uncertainty associated with the measurement increases.

With ADS-B, the measurements of the target states are conducted by a sensor that is not on-board the own-ship. Once the measurement is taken, it must be processed and then transmitted to the own-ship. The time that elapses between taking the measurement and using of that information is called latency.

The total latency present in information depends on the systems that the information encountered before it is used on-board the own-ship. In the case where the own-ship received ADS-B messages directly from the target, total latency consists of the latency that the avionics introduce on-board the target and the own-ship. If the information was received via ADS-R or TIS-B, total latency also includes the latency added by the ground systems.

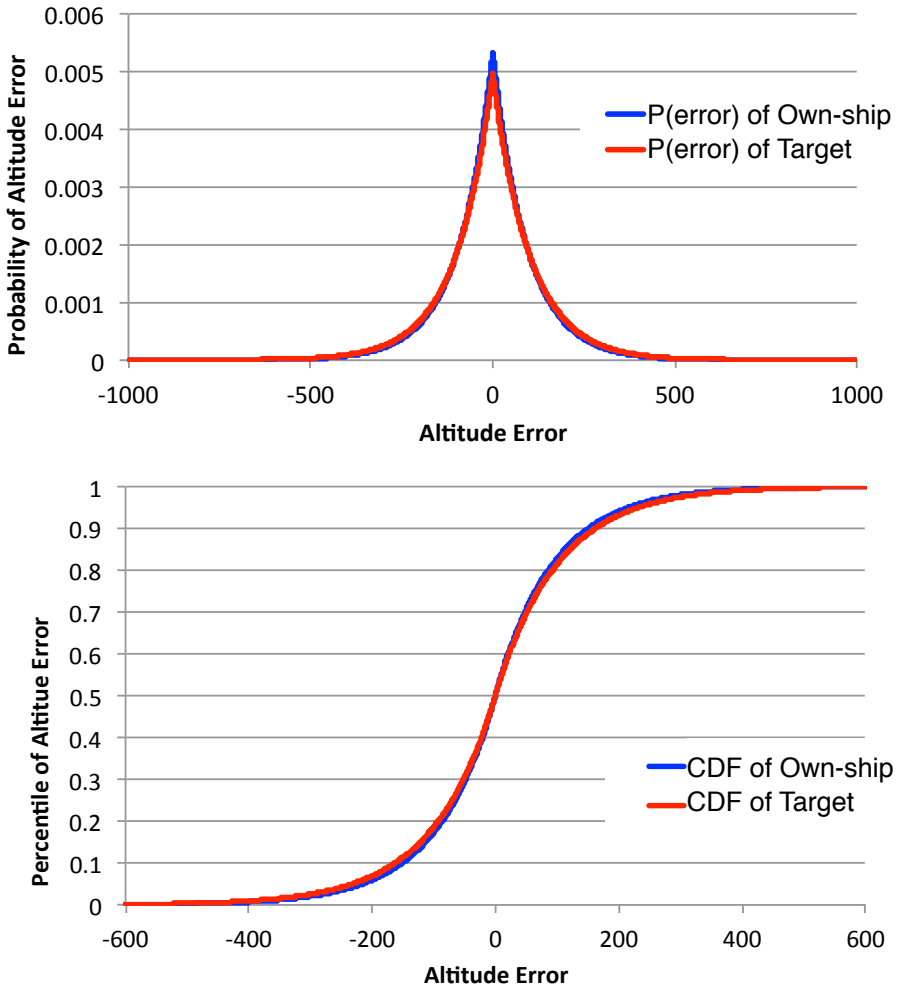


Figure 5-15: Laplace Distribution Used to Simulate Own-Ship and Target Altimetry Errors at Altitudes in Excess of 41,000 ft

Generally, the amount of latency introduced by a given system is known. As a result, standards require those systems to compensate the position information for latency. The compensation extrapolates the received position linearly for a nominal amount of time. This compensation is the source of error due to latency. Figure 5-16 shows a representation of how this error is introduced. The left-hand side of the figure shows the aircraft at the moment when the measurement is taken. By the time the information is actually used, the aircraft have moved to the right-hand side. The black lines are the true tracks that the aircraft traveled in between. The gray tracks are the tracks approximated by the extrapolation used for the latency compensation. Thus, linear extrapolation for a set amount

of time can introduce error in three ways. First, if the amount of time used in the extrapolation is not accurate, over- or under-compensation in the direction of flight may occur ("compensation time error"). Second, a linear extrapolation does not take into account maneuvers that occurred during that time, resulting in a cross-track error ("maneuver error"). Last, since the compensation uses the velocity information in the ADS-B message, error in the velocity estimate may cause over- or under-compensation in the direction of flight as well ("error due to velocity uncertainty").

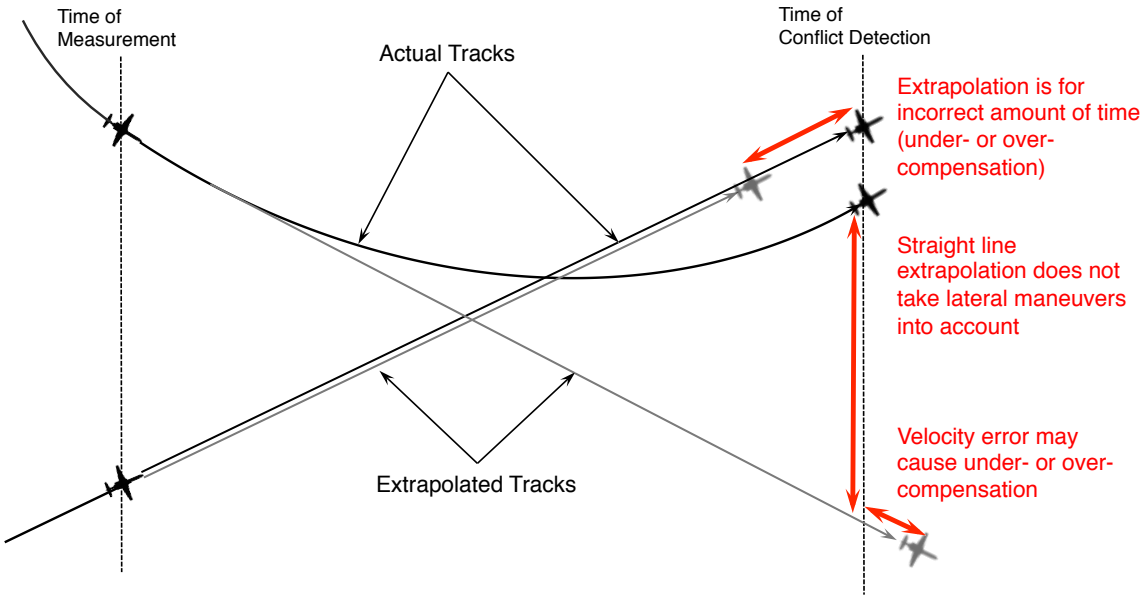


Figure 5-16: Schematic Representation of Error Sources Introduced by Latency Compensation

The sum of these three errors results in an elliptical error distribution around the true position of the aircraft at the time the information is used. Figure 5-17 shows the magnitude of each of the three sources of error and Figure 5-18 shows the sum and geometric distribution of a 6 second latency compensation error for an aircraft flying at 120kts.

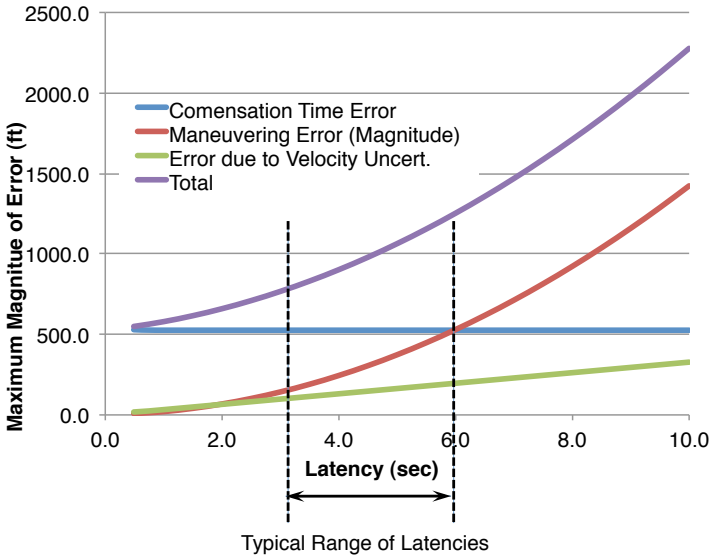


Figure 5-17: Error Due to Latency Compensation as a Function of Growing Latency

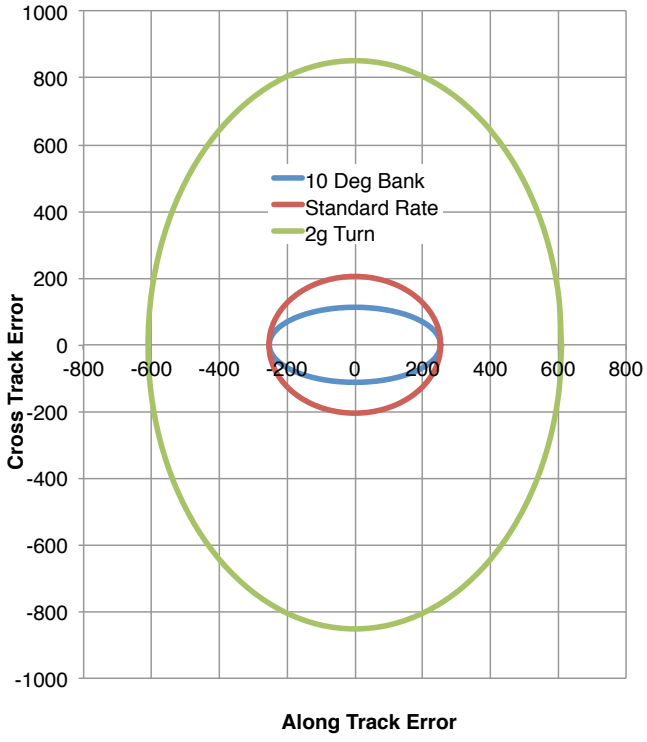


Figure 5-18: Sample Latency Error Ellipses for 6-second Latency Compensation

The model for errors due to latency consisted of a latency trajectory generator that takes the following states as input:

- Original flight track from encounter set
- Aircraft velocity with and without error
- Total compensated latency (second)
- Latency compensation error (seconds)

The total compensated latency remains constant throughout an encounter and is specific to the type of target (ADS-B vs. ADS-R vs. TIS-B). The latency compensation error defines the sigma for a Gaussian distribution that is sampled once per state update and then added to the total compensated latency. Together, they represent the total time compensated by the constant heading extrapolation (t_{TOT}).

Given the total amount of time to be compensated, a latency trajectory is generated. The latency trajectory originates at the current time minus the time to be extrapolated and uses the velocity with error to propagate a constant heading t_{TOT} into the future. The error due to latency compensation is the difference between the true position and the position that the latency trajectory predicted. This difference is the error that is added as a position error to the state data. Figure 5-19 shows a time evolution of the latency error. During this particular encounter, the target initiates a turn at t = 140 seconds, causing the cross track error to increase.

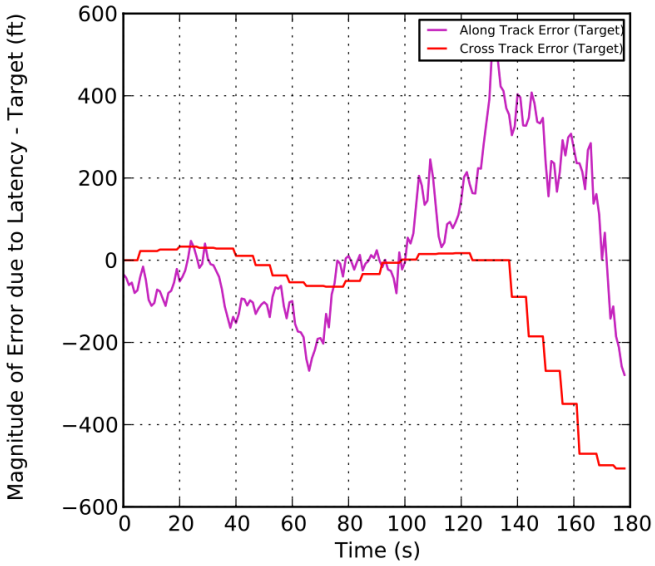


Figure 5-19: Sample Latency Error Separated into Cross Track (red) and Along Track (pink) Components

ERROR DUE TO REDUCED PROBABILITY OF RECEPTION

Similarly, errors also are introduced by the low rates at which new information about a given target becomes available. An update on position, velocity, and altitude only becomes available once per rotation for radars that rotate once every 4.2 seconds in the terminal area and once every 12 seconds in the en-route environment. In high-density ADS-R environments, the ground system may enter a mode of "graceful degradation" where the rates at which cross link transmissions are re-broadcast at a reduced rate to reduce frequency congestion. Once the message reaches the own-ship, incorrect decoding or message overlaps may cause TIS-B, ADS-R, and aircraft-to-aircraft ADS-B messages to be dropped intermittently

On-board the own-ship, however, trackers such as the one defined in DO-317A provide updates to applications such as TSAA once per second. To do so, the tracker internally extrapolates the position information, introducing errors in a pattern resembling latency error introduction.

As described in the Surveillance and Broadcast Services Description Document [64], each of the services has a nominal rate at which an update can be expected. Table 5-2 shows the interval and probability that an update is received within that interval. To model the probability that a given update is in fact received by the own-ship, a random sample is drawn from a uniform distribution. If the sample is above the threshold specific to the interval that is to be modeled, the message is received (thresholds also shown in Table 5-2). Figure 5-20 shows the probability density function and the cumulative density function for an update interval of 6 seconds.

Table 5-2: Update Intervals for ADS-B, ADS-R and TIS-B

Environment		Interval (sec)	Cumulative Probability	Sampling Threshold
ADS-B	Terminal and En-Route (High-Update)	3	95%	0.624
	En-Route, (Low-Update)	6	95%	0.391
ADS-R	Terminal	5	95%	0.451
	En-Route	10	95%	0.255
TIS-B	Terminal	6	95%	0.291
	En-Route	12.1	95%	0.213

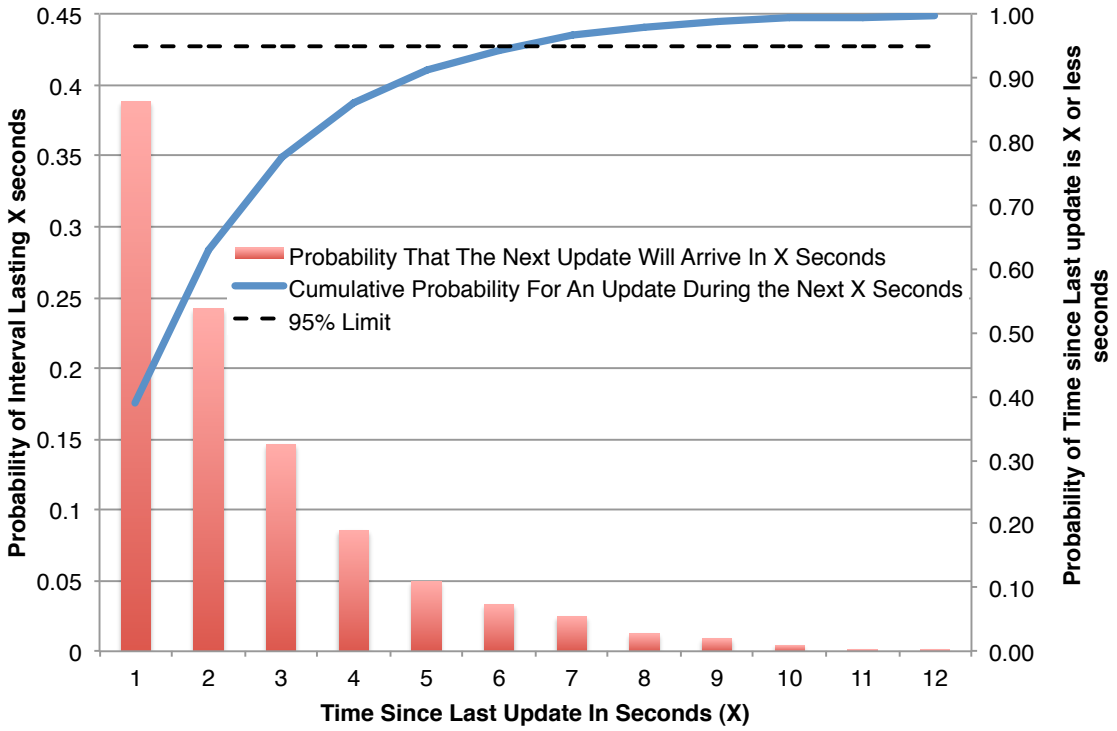


Figure 5-20: Probability Density Function and Cumulative Probability for 95%, 6 Seconds Update Interval

5.5.3 MODEL PARAMETERS USED FOR ADS-B, ADS-R AND TIS-B TARGETS

The models above were implemented such that they could be adjusted to define any desired target type and quality. The model assumptions and parameters are summarized here.

Two important distinctions must be made. First, standards defining the performance requirements for ADS-B have evolved over time, and the requirements for position/velocity accuracy and latency have evolved along with them. The values presented here are specific to the most recent version of those standards (DO-260B and DO-282B) and not their earlier versions. Second, the error numbers shown here are the maximum allowable values and represent worst-case scenarios. In reality, the errors actually present during real-world operations are expected to be significantly less than the maximum values shown here.

PARAMETERS FOR ADS-B TARGETS

Figure 5-21 shows the data flow for ADS-B targets. The measurement sensor is assumed to be a GNSS engine. The top of the figure depicts the maximum allowable compensated latency and the he bottom lists the maximum allowable latency compensation error. Note

that the sum of the compensation errors contributed by the various systems is added using the root-mean square approach, as these are the sigma values of Gaussian distributions.

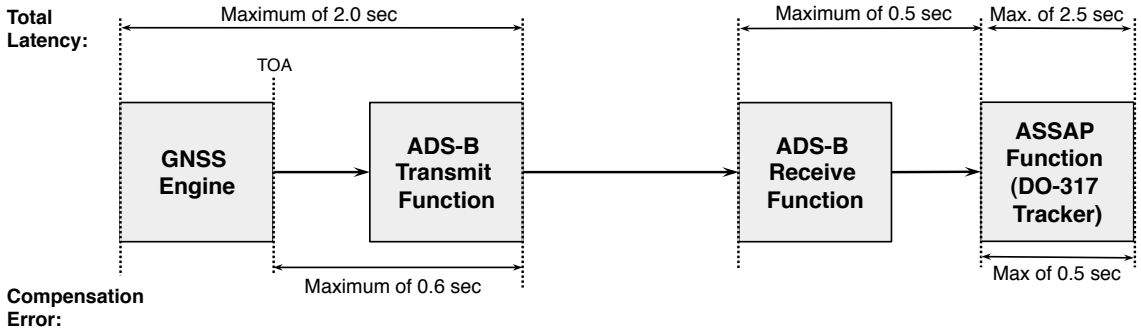


Figure 5-21: System Components, Data Flow and Latency Sources for ADS-B Targets

Table 5-3 shows the settings used to simulate ADS-B targets using the models introduced above. Since the compensation errors represent the sigma values for Gaussian distributions, they are combined using the root mean square. Message travel times would increase as distance increases but are assumed to be negligible given that they travel at the speed of light.

Table 5-3: Simulation Model Settings for ADS-B Targets

Model Parameter	Model Parameter Setting	Resulting Model Behavior
Position Error Auto-Correlation	300 sec	5 minute auto correlation time
Velocity Error Auto-Correlation	300 sec	
Position and Velocity Error Shape Factor (Covariance)	2.448	Worst-case circular error shape
Position Error Bound (NACp)	> 5	Maximum error allowed by ATSA-ARIB
Velocity Error Bound (NACv)	> 1	
Total Compensated Latency	5 sec	Total compensated latency allowed by §91.227, DO-242, DO-260B and DO-317
Latency Compensation Error	0.78 sec	
Update Rate	0.682	3 second update rate, 95% probability

PARAMETERS FOR ADS-R TARGETS

Figure 5-22 shows the data flow for ADS-R targets. Again, the measurement sensor is assumed to be a GNSS engine. The additional compensated latency is 1 second when compared to an ADS-B target and the additional latency compensation error is 0.1 seconds.

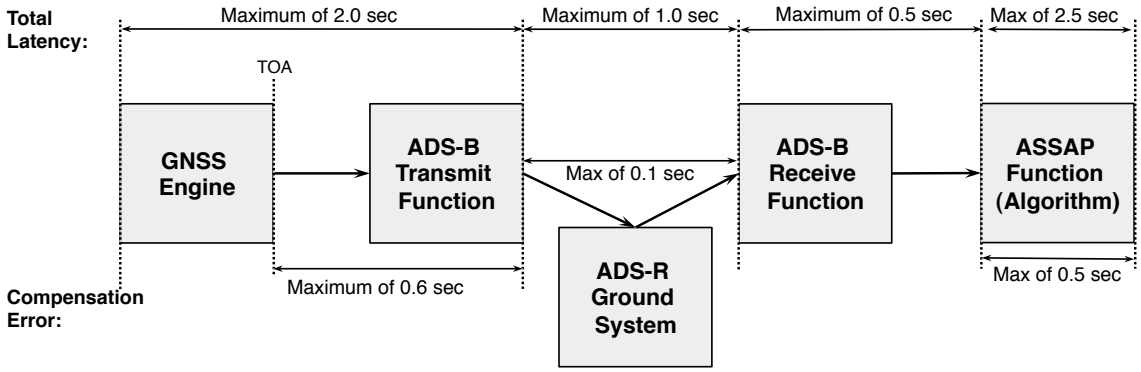


Figure 5-22: System Components, Data Flow, and Latency Sources for ADS-R Targets

Table 5-3 shows which settings are different from the setting used for the ADS-B target. All others remain the same. Again, message travel times are assumed to be negligible.

Table 5-4: Simulation Model Settings for ADS-R Targets

Model Parameter	Model Parameter Setting	Resulting Model Behavior
Total Compensated Latency	6 sec	Total compensated latency allowed by §91.227, DO-242, DO-260B and DO-317
Latency Compensation Error	0.79sec	
Update Rate	0.451	5 second update rate, 95% probability

PARAMETERS FOR TIS-B TARGETS

Figure 5-23 shows the data flow for TIS-B targets. The measurement sensor is assumed to be a ground based radar system. Compared to GNSS engines, the rate at which new information about a TIS-B target becomes available is less straightforward. At minimum, updates become available at the radar's rotation rate, assuming a probability of detection by the radar of 1. In the US, ASR-11 radars in the terminal area rotate once per 4.8 seconds while long range ARSR-4 radars in the en-route environment rotate once every 12 seconds. However, a target may be in view of more than a single radar. As a result, updates about a given target become available more frequently than the rotation rate of a single radar, though they may not arrive in even intervals.

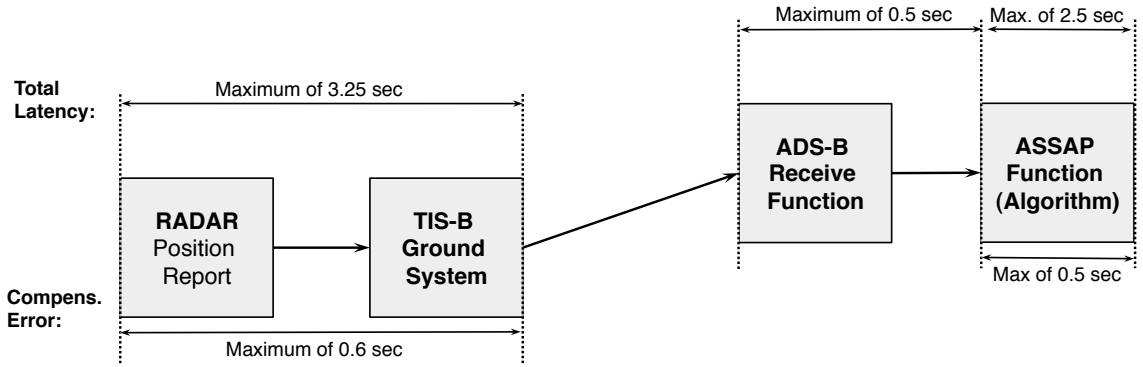


Figure 5-23: System Components, Data Flow and Latency Sources for TIS-B Targets

Table 5-3 shows the settings used to simulate TIS-B targets. Again, message travel times are assumed to be negligible. Though ATSA-AIRB requires NACv values to 1 or greater, TIS-B targets may report NACv values of 0. A NACv value of 0 is assumed to be equivalent to 30m/s of error.

Table 5-5: Simulation Model Settings for TIS-B Targets

Model Parameter	Model Parameter Setting	Resulting Model Behavior
Position Error Auto-Correlation	98 sec	98 and 45 second auto correlation time
Velocity Error Auto-Correlation	45 sec	
Position and Velocity Error Shape Factor (Covariance)	2.1	Elliptical error shape for radar
Position Error Bound (NACp)	> 5	Maximum error allowed by ATSA-ARIB (see note in text)
Velocity Error Bound (NACv)	> 1	
Total Compensated Latency	6.25 sec	Total compensated latency allowed by §91.227, DO-242, DO-260B and DO-317
Latency Compensation Error	0.78 sec	
Update Rate	0.391	6 second update rate, 95% probability

PARAMETERS FOR OWN-SHIP

Lastly, Figure 5-24 shows the data flow for the TSAA own-ship. The own-ship is assumed to have a GNSS engine similar to the ADS-B target above but with a 1 second update rate and 100% probability of reception. Table 5-3 shows the settings used to simulate the own-ship. It should be noted that, even though the own-ship state data flows through the sample ASSAP tracker, it is not smoothed by the filters the same way the target data is smoothed.

Instead, the sample tracker passes the own-ship data as point estimates without attempting to take out any noise or bias that might be present.

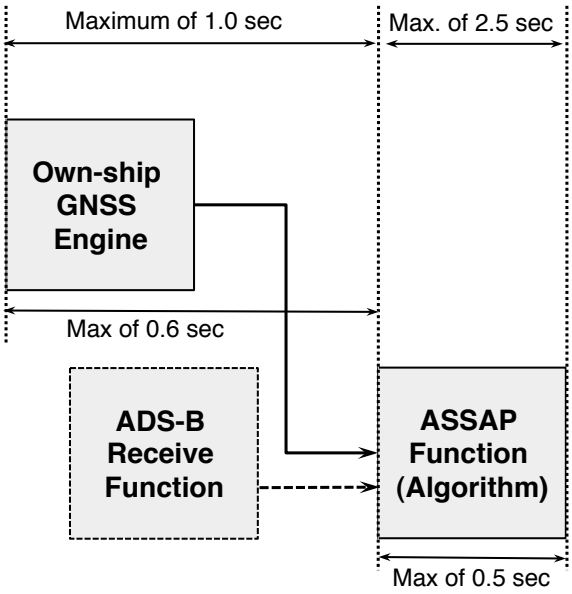


Figure 5-24: System Components, Data Flow and Latency Sources for the TSAA Own-ship

Table 5-6: Simulation Model Settings for TSAA Own-ship

Model Parameter	Model Parameter Setting	Resulting Model Behavior
Position Error Auto-Correlation	300 sec	5 minute auto correlation time
Velocity Error Auto-Correlation	300 sec	
Position and Velocity Error Shape Factor (Covariance)	2.448	Worst-case circular error shape
Position Error Bound (NACp)	> 5	Maximum error allowed by ATSA-ARIB
Velocity Error Bound (NACv)	> 1	
Total Compensated Latency	3.5 sec	Total compensated latency allowed by §91.227, DO-242, DO-260B and DO-317
Latency Compensation Error	0.78 sec	
Update Rate	1	1 second update rate, 100% probability

5.5.4 ALERTING STATISTICS ANALYZER

The alerting statistics analyzer evaluates the alerting system's performance for a given encounter data set, level of uncertainty, and set of algorithm parameters by applying the

alert scoring approach introduced in section 3.4. During the initial phase of algorithm development (Figure 5-6), the performance metrics were calculated based on all encounters in the LLEM Master Encounter Set. During the second phase, however, the metrics were calculated using the Low Altitude And Airport Operations Master Encounter Set. Given their limitations, the scripted encounters were excluded from the calculation of the nuisance alert rate and the average alert time. To ensure that those performance metrics were representative of real world alerting performance, they were calculated based on the encounters derived from SBS surveillance data only. Table 5-7 shows the mapping of which encounters were used to calculate which performance metrics.

Table 5-7: Mapping of what Encounters in the Low Altitude and Airport Ops Data Set Were Used to Calculate Performance Metrics

Performance Metric	Encounter Used
Nuisance Alert Rate	SBS Encounters
Missed Alert Percentage	SBS Encounters and Scripted Encounters
Late Alert Percentage	SBS Encounters and Scripted Encounters
Average Time of Alert Before CPA	SBS Encounters
Total Alert Count	SBS Encounters and Scripted Encounters

MULTIPLE CPA SCORING

One challenge of evaluating encounters from the Low Altitude and Airport Operations Master Encounter Data Set was that in some cases the own-ship encounters certain targets more than once over the duration of a single own-ship trip. Additionally, especially during operations in the airport patterns, aircraft frequently remained in close proximity for a prolonged amount of time. Though there is a single, "global" closest point of approach (CPA) for the particular encounter, in both cases the two aircraft effectively re-encounter each other multiple times, generating multiple local CPAs. This results in three challenges when scoring alerts. First, if the system alerts on a local CPA and then re-alerts on the global CPA, the second alert may be scored as late incorrectly. Second, if there is no re-alert on the global CPA but the alerting system simply remains in alert state, the non-alert may be scored as a missed alert incorrectly. Alternatively, the alert from the initial local CPA could be used for scoring but it may increase the average alert time artificially. Third, any alert that is issued on a local CPA after the global CPA would be scored as a nuisance alert incorrectly. Highlighted in Figure 5-25 is a case where a PAZ alert, indicated by a star, and a CAZ alert, indicated by a filled circle, were scored incorrectly as a late alert. Also highlighted

are three PAZ alerts that were scored as nuisance alerts since they were issued after the global CPA.

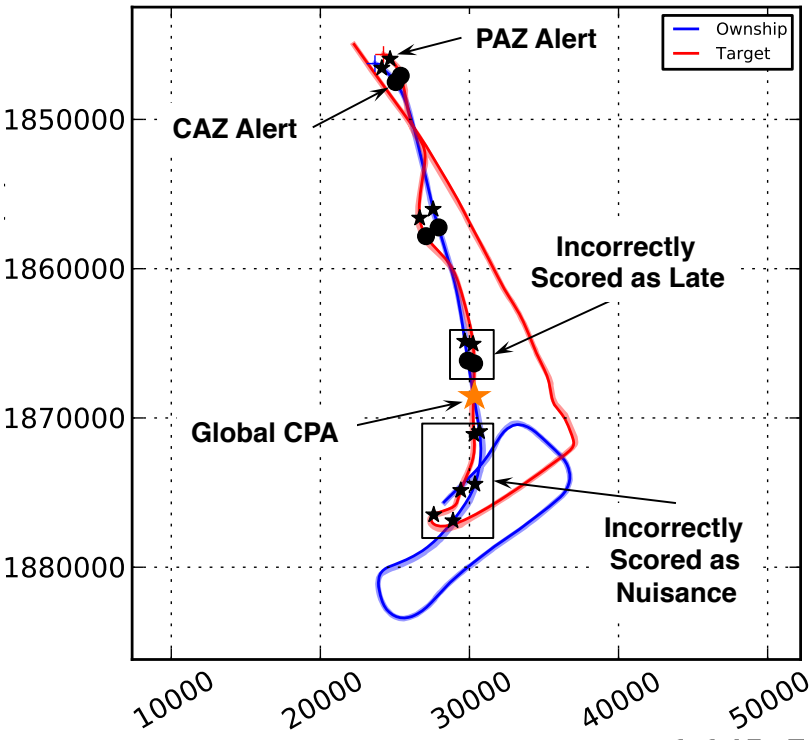


Figure 5-25: Sample Prolonged Proximity Encounter

To address this, the scoring method was adjusted to score alerts with respect to the closest local CPAs instead of a single, global CPA. A local CPA was defined as the CPA of any period of time when the target at least penetrated the may-alert zone. Once the target returns to the Non-Hazard Zone, it is considered a new target and any alert thereafter will be evaluated against a new, future local CPA. If an alert is issued before the target penetrates the may-alert zone, the target must penetrate the may-alert zone within the next 60 seconds for the alert to not be considered a nuisance alert. Alerts issued after the CPA are not considered nuisance alerts if they were issued while the target was still in the may-alert zone.

5.6 Step 3: Analysis of Algorithm Behavior and Visualization of Performance in the Performance Space

Every point in the performance space is specific to a particular algorithm parameter combination as well as with the data set used and the level of uncertainty introduced by the simulation. Therefore, a point's location in the performance space is determined primarily by the algorithm parameters but also is affected by the data set and data quality used to calculate the algorithm performance. Referring to Figure 5-3, the last step in the algorithm tuning procedure is to evaluate the performance of the various algorithm parameter combinations in the performance space in to select a final, tuned algorithm parameter combination.

The tools to determine this final, tuned algorithm parameter combination are presented in this section. First, two visualization tools are introduced; second, the analytical process used to identify the final settings of the algorithm parameters is discussed.

5.6.1 VISUALIZATION TOOLS TO VISUALIZE THE PERFORMANCE SPACE

The performance metrics that define the performance space for TSAA were identified in section 3.4 and are listed below.

- Nuisance Alert Rate
- Missed Alert Percentage
- Late Alert Percentage
- Average Time of Alert Before CPA
- Total Alert Count

Two methods of visualizing the performance space were identified to be valuable during the evaluation of the TSAA algorithm: The Radar Chart Visualization and a scatter plot adaptation of Kuchar's ROC method approach.

RADAR CHART VISUALIZATION

The radar chart plots each metric as a spoke on a radial plot. Figure 5-26 shows a sample radar plot of two fictional alerting systems. To make this visualization tool useful for analyzing TSAA, the spokes were all normalized to range from 0 to 1, which was necessary for two reasons. First, not all performance metrics occupy the same range. For example, the missed alert percentages range from 0 to 5% where the total alert count may range from 3,000 to 25,000. Therefore, to be able to plot both metrics on the same plot without losing the precision at lower absolute number differences (i.e., 0.05 vs. 17,000), both numbers were normalized to 1. Second, different performance metrics desire different directions on the spoke; for example, missed alerts should be minimized while the average time of alert

before CPA should be maximized. For TSAA, smaller values were defined to be better, which therefore requires some of the normalization values to be flipped. Table 5-8 shows the normalization parameters as well as which spokes had reversed normalization directions.

Table 5-8: Normalization Values used for Polar Chart Visualization

Performance Metric	Zero – Equivalent	One-Equivalent	Reversed
Nuisance Alert Rate (alerts per hour)	0	0.25	No
Missed Alert Percentage	0	12	No
Late Alert Percentage	0	8	No
Average Time of Alert Before CPA (seconds)	35	20	Yes
Total Alert Count	2500	20000	No

In Figure 5-26, the green system outperforms the blue system in all of the performance metrics.

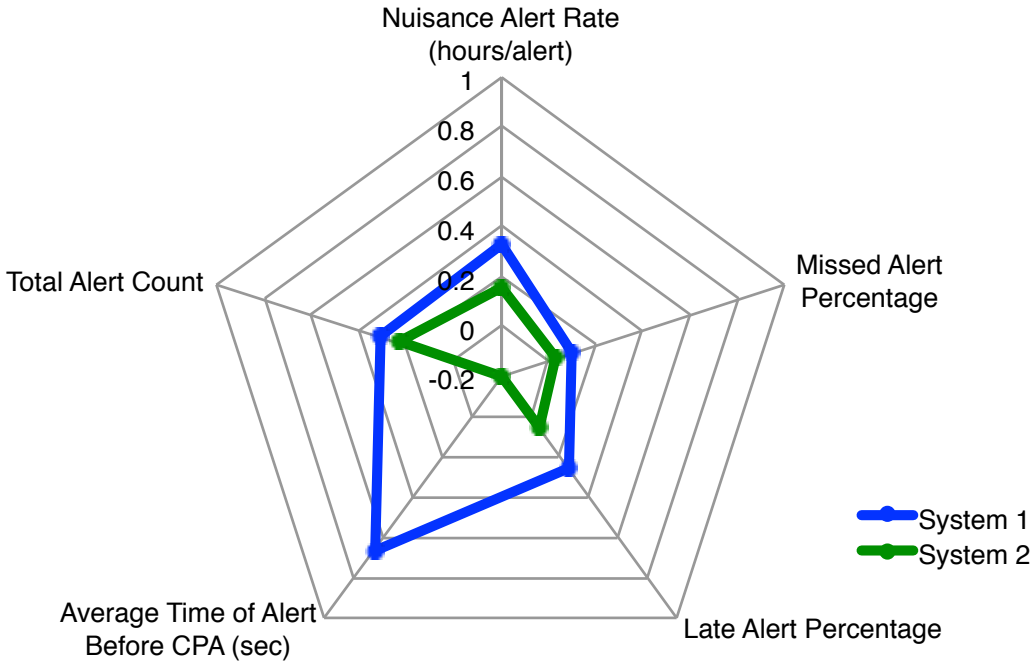


Figure 5-26: Sample Radar Chart Visualization for Two Algorithm Parameter Combinations (smaller is better)

The radar chart visualization lends itself well to comparing the performance of a small number of algorithm implementations. However, a scatter plot adaptation of Kuchar's ROC method is much better suited to visualizing the performances of a large set of algorithm parameter combinations.

SCATTER PLOT ADAPTATION OF KUCHAR'S ROC METHOD

Kuchar proposed a unified method to evaluate performance trade-offs for hazard alerting systems [10]. Using the receiver operating characteristic curve (ROC) concept from signal detection theory, an alerting system can be evaluated by plotting the probability of a correct detection vs. the probability of a nuisance alert (Figure 5-27). The optimal performance point is the top left corner with a 100% probability of correct detection and 0% nuisance alerts. For a particular alerting system, selecting different alerting thresholds results in different combinations of correct detection and nuisance alert probabilities, which is manifested in movements along the ROC curve.

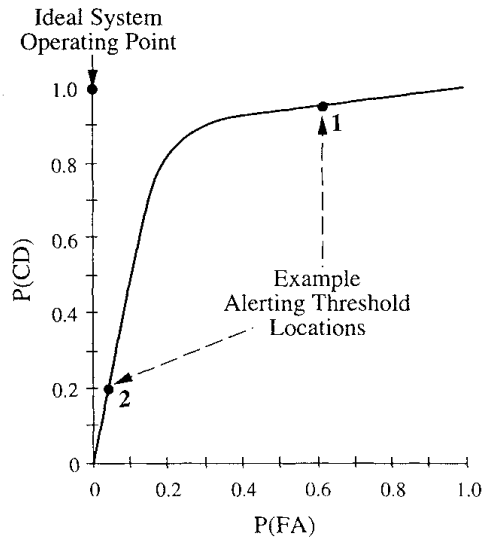


Figure 5-27: Sample Receiver Operating Curve Adapted for Conflict Alerting System Performance Evaluation (Reproduced from [10]**)**

The ROC curve also illustrates how performance metrics trade against each other. In general, earlier or more frequent alerting decreases the likelihood of a missed alert but increases the likelihood of a nuisance alert.

This method was adapted during the development of TSAA. As shown in Figure 5-27, Kuchar notionally uses two alerting thresholds that lie along the curve. In the case of TSAA, a large number of parameter combinations, which in effect represent different alerting

thresholds, will be visualized as individual dots representing a single performance point each. Using the performance metrics defined for TSAA, the percent correct detection is calculated as shown in Equation 6.

Equation 6: Calculation of Probability Correct Detection for TSAA

$$P(Correct\ Detection) = 1 - P(Late\ Alert) - P(Missed\ Alert)$$

In its original form the ROC method uses the probability of false alarm on the x-axis. For the purposes of TSAA, only nuisance alerts will be evaluated and false alarms are excluded. Therefore, the x-axis will be re-defined as the number of nuisance alerts per hour. In order to show the remaining two metrics, the method was further adjusted to show the average alert time (color of the dot) and the total number of issued alerts (size of dot). Figure 5-28 shows the modified plot for 100 different algorithm parameter combinations that originally were generated by the Latin hypercube approach described in section 5.4.1. The black hexagon also shows the performance of a basic TAS (i.e., TCAS I algorithm) implementation that is compliant with the TCAS I algorithm standard (DO-197). AAT stands for the average alert time of the TAS implementation.

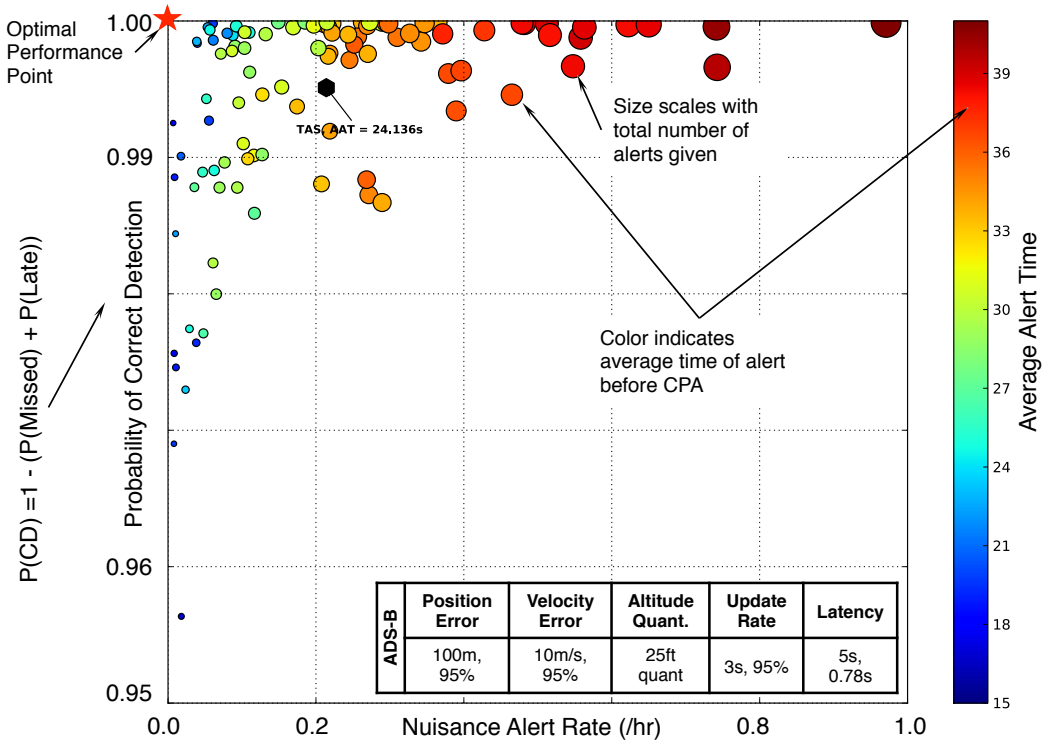


Figure 5-28: PCD vs. Nuisance Rate visualization for 100 different algorithm parameter combinations

Given the single plot shown in Figure 5-28, an analyst can evaluate the trade-offs between the various performance metrics for a large set of different algorithm implementations. Effectively, the desired outcome of tuning the algorithm is to obtain a parameter combination with a performance represented by a small, warm-colored dot in the top left corner of the plot.

One concept that comes to mind when evaluating Figure 5-28 is the Pareto front. However, fitting a Pareto front through the points in Figure 5-28 may be misleading. Since the visualization reduces a five-dimensional performance space to two dimensions spatially, a Pareto front in Figure 5-28 would only show the systems probability of correct detection to nuisance alert rate trade-off and where its optimal design points are. However, it does not provide any information from the perspective of the other performance metrics, such as average alert time.

5.6.2 GENERATION OF HIGH ORDER MODEL REPRESENTATION BASED ON MULTIVARIATE PERFORMANCE DATA

As discussed above, it is difficult to define a single definition of utility that combines all performance metrics into a single. Instead, this development process took a direct approach to evaluating the trade-offs relevant to each performance metrics. Fitting an HDMR to the performance data in the performance space enabled this approach.

HDMR is used to identify the global sensitivity of a particular performance metric to the different input parameters. In Equation 7, the performance metric of interest is represented by the vector *f(x)* and is expresses as a combination a constant f_0, the sum of first and higher order interactions.

Equation 7: Formula Used by High Dimensional Model Representation (HDMR) Method

$$f(\mathbf{x}) = f_0 + \sum_{i=1}^{n} f_i(x_i) + \sum_{1 \le i < j \le n} f_{ij}(x_i, x_j) + \ldots + f_{12\ldots n}(x_1, x_2, \ldots, x_n)$$

In order to express the performance values in the form of Equation 7, the *f* values must first be determined. As shown in reference [95], the values can be calculated as follows:

Equation 8: Calculation of *f*-Coefficients for HDMR Model Fitting

$$f_0 = E\big(f(x)\big)$$
$$f_i = E(f(x)|x_i) - E\big(f(x)\big)$$
$$f_{ij} = E\big(f(x)\big|x_i, x_j\big) - f_i - f_j - E\big(f(x)\big)$$

Once the model has been fit, the surrogate can be used to identify which input parameters contribute most significantly to the variations in *f(x)*. In other words, the sensitivity of the performance metrics to the individual input parameters can be evaluated using Equation 9. As with other regression approaches, the R^2 should be evaluated in order to determine how well the model actually fits the data.

Equation 9: First Order Sensitivity Equitation

$$S_i = \frac{V[E(Y|X_i)]}{V(Y)} = \frac{V[f_i(x_i)]}{V(Y)}$$

Again, there could be higher order interactions between the parameters. As a result, the sensitivity indices of the performance metrics for second and higher order combinations between the parameters are also calculated. Equation 10 is used to calculate second order sensitivity indices.

Equation 10: Second Order Sensitivity Equitation

$$S_{ij} = \frac{V\left[E(Y|X_i, X_j)\right]}{V(Y)}$$

As discussed in more detail in reference [95], the sum of all sensitivity indices sums up to 1.

A software package that implements these equations was used to fit the HDMR model to the TSAA performance data [99]. As inputs, it requires the normalized parameter settings used to generate the performance data and the resulting TSAA performance. As outputs, it identifies how much of the variability in a given performance metric can be attributed to the variations in each parameter (sensitivity). It also allows for the more in-depth analysis of the first and second order trade-offs between parameters and performance metrics.

STEPS TO SELECT FINAL PARAMETER SETTINGS

Using the visualization tools and the analytical identification of high-impact parameters with the HDMR analysis, the final parameter settings are selected by following these steps:

1. **Fit HDMR model to performance data:** Given the algorithm combinations sampled from the parameter hyperspace, simulate one encounter data set for all combinations and calculate the resulting performance. Fit the HDMR model to the data.
2. **Identify the high-impact parameters using the results from the HDMR model:** Identify which parameters account for the largest percentage of variability in the

performance metrics, i.e., the parameters to which the performance metrics are most sensitive.

3. **Evaluate trade-offs for the high-impact parameters:** Given the high-impact parameters, evaluate their trade-offs to gain an understanding of the trade-space. Identify any "break-points" where there is a sharp change in performance.
4. **Select the setting to obtain the performance required by the system requirements:** Once the trade-offs are understood, select the parameter values such that the performance requirements set out in the system requirements can be met or approximated as closely as possible.
5. **Evaluate trade-offs and select settings for the non-high-impact parameters:** Algorithm parameters that do not significantly impact performance should be selected in light of other system requirements or engineering considerations such as computational load or operational insights.
6. **Verify resulting performance in comparison against the original performance data:** Re-run the same data set used in step 1 to calculate the performance of the tuned parameters.

Chapter 6

APPLICATION OF PERFORMANCE EVALUATION AND TUNING METHOD TO SAMPLE TSAA ALGORITHM

The parameters of the exemplar TSAA algorithm were tuned using the method described in the previous chapter as shown in Figure 6-1.

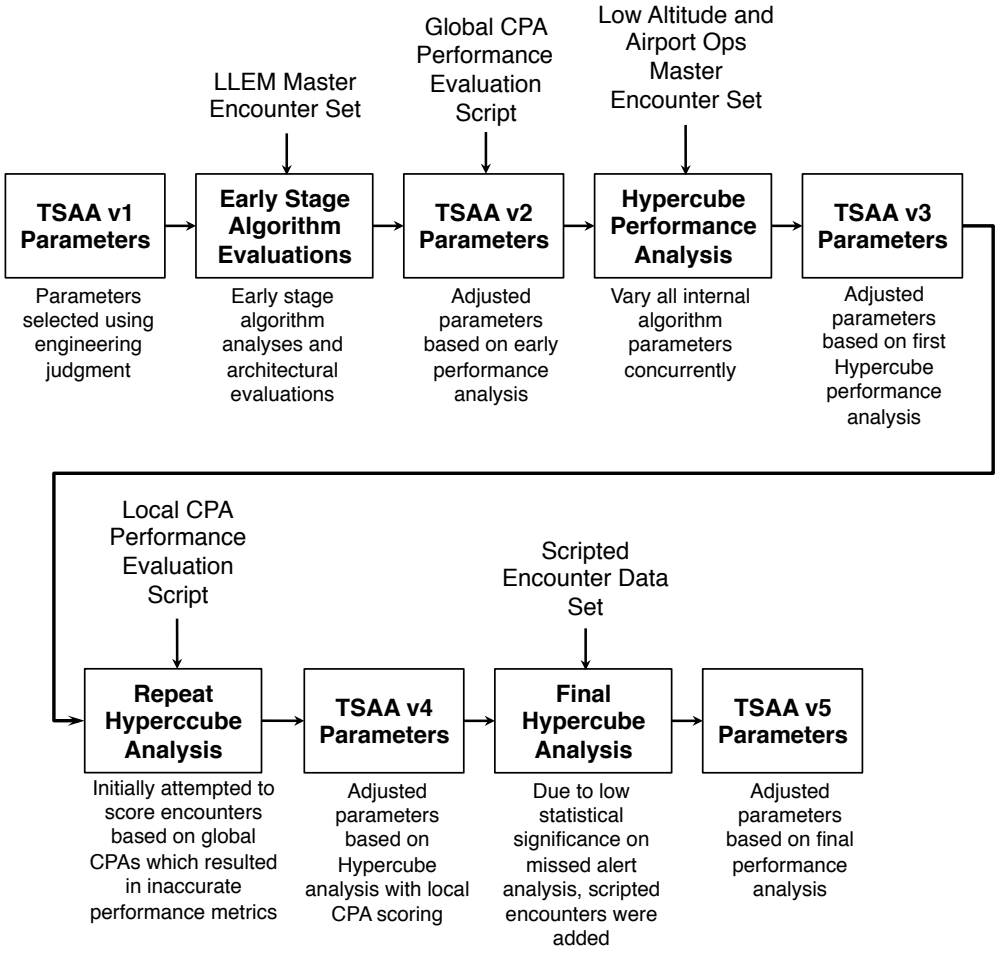


Figure 6-1: TSAA Parameter Version Evolution

The process of tuning the algorithm parameters was iterative and benefited from learned lessons along the way. Figure 6-1 also shows which data set and scoring method was used at which step. This chapter gives a more detailed description of the process of the selecting the version 5 TSAA parameters. Note that this tuning occurred using the TSAA implementation that did not contain the DO-317A tracker; as such, TSAA's performance with the tracker would be expected to improve as a result of the reduced levels of error in the data.

Throughout the tuning process, David Elliott and Douglas Havens at the MITRE Corporation provided invaluable support. In parallel to the analysis conducted at MIT, they also conducted an independent analysis of the TSAA algorithm and its performance. Though they used a slightly different simulation environment, their results generally agreed with the results obtained at MIT within 1%.

6.1 Simulation of Encounters and Generation of Performance Data

First, the setup and execution of each step in the tuning method shown in Figure 5-3 is described separately below.

6.1.1 Generating Parameter Combinations using the Latin Hypercube Method

Table 6-1 shows the internal TSAA parameters and the ranges over which they were evaluated. As mentioned, six of the 14 parameters were fixed based on independent observations and analysis. Fixing these five parameters has the additional benefit of reducing the dimensionality of the TSAA hypercube:

Table 6-1: Adjustable Parameter Internal to the Prototype TSAA Algorithm

	Algorithm Internal Parameter	Setting	
Adjustable Parameters	Look-ahead Time (s)	10 sec	60 sec
	Trajectory Discretization (s)	0.1 sec	5 sec
	Turn Rate Filter (#)	2 updates	15 updates
	Vertical Rate Filter (#)	2 updates	15 updates
	Min. PAZ Radius (ft)	500 ft	1000 ft
	Min. PAZ Height (ft)	200 ft	1000 ft
	Hor. PAZ Scaling (s)	0 sec	20 sec
	Vert. PAZ Scaling (s)	0 sec	10 sec

Preset Parameters	Conflict Search Frequency (s)	Once per second
	Target Discontinuation Threshold (s)	15 sec
	CAZ Radius (ft)	500ft
	CAZ Height (ft)	±200ft
	Re-alert Delay (s)	6 sec
	Double Trigger	15 sec

Using the Latin hypercube method, 100 hypercube points were generated using the ranges defined above.

6.1.2 CONFIGURING THE SIMULATION TOOL: DATA SETS AND NOMINAL TARGETS

Given the system requirement for TSAA to operate in the airport environment (Requirement FH1), the final tuning of the TSAA algorithm parameters was performed using the Low Altitude and Airport Operations Master Encounter Set. Since the airport environment is a more challenging location to perform alerting than the en-route environment, it is assumed that if TSAA performs acceptably in the airport environment, it will do so in the en-route environment as well. Additionally, the Low Altitude Master Encounter Set was generated from the highest density operations in the US NAS and thus represents a very challenging environment for TSAA. The likelihood of a flight crew only operating in this type of environment is low.

Certain algorithm parameters may affect the robustness of the TSAA algorithm against varying levels of state uncertainty. To ensure that any type of interaction between such parameters and the levels of error both are captured, four nominal targets were defined that represented targets that would be encountered commonly during operations in the NAS. The ADS-B nominal target represents an ADS-B equipped target compliant with the US ADS-B Out rule. The ADS-R target is the same as the ADS-B target but on the opposing ADS-B frequency. The TIS-B1 target is tracked by a single ground based radar system found frequently around high-density airports, such as an ASR-11. As such, the TIS-B1 target has a high update rate (radar rotation rate of once every 4.8 seconds) and a low level of position error. The TIS-B2 target is defined as a worst-case TIS-B target. It is modeled as an aircraft in view of a single, long-range radar such as an ARSR-4 with an update rate of once every 12 seconds and a significant level of error in its state data. Table 6-2 shows a summary of the error parameters used to configure the error models described in the encounter simulator. The performance for the 100 algorithm parameter combinations was evaluated for each of the four nominal targets.

Table 6-2: Error Parameters for Nominal Targets

Nominal Target Type	Position Error	Velocity Error	Altitude Quantization	Update Rate	Latency Compensation
ADS-B	NACp = 8	NACv = 1	25ft	3sec, 95%	5s, 0.78s
ADS-R	NACp = 8	NACv = 1	25ft	5sec, 95%	6s, 0.78s
TIS-B1	NACp = 8	NACv = 1	25ft	6sec, 95%	6.25s, 0.78s
TIS-B2	NACp = 5	NACv = 0	100ft	12sec, 95%	6.25s, 0.78s

6.1.3 SIZE OF SCORING ZONES USED FOR PERFORMANCE EVALUATION

Since the Low Altitude and Airport Operations Master Encounters Set consists mainly of terminal environment operations, the Hazard Zone and Non-Hazard Zone sizes for the terminal environment were used (Table 6-3).

Table 6-3: Terminal Area Hazard and Non-Hazard Zones used for TSAA Algorithm Tuning With The Low Altitude and Airport Operations Master Encounter Set

	Hazard Zone		Non-Hazard Zone	
Environment	**Vert.**	**Hor.**	**Vert.**	**Hor.**
Terminal	200ft	500ft	500ft	½ NM

6.2 Final Selection of TSAA Algorithm Parameters

The following sections follow the approach outlined in section 5.6.2.

6.2.1 DATA GENERATION AND HDMR MODEL FITTING

All 100 hypercube points were evaluated for each of the nominal targets defined in Table 6-2. Figure 6-2 through Figure 6-5 show the performance scatter plot for all 100 points for each one of the nominal targets. Refer to section 5.6.1 for a more detailed instruction on the visualization approach.

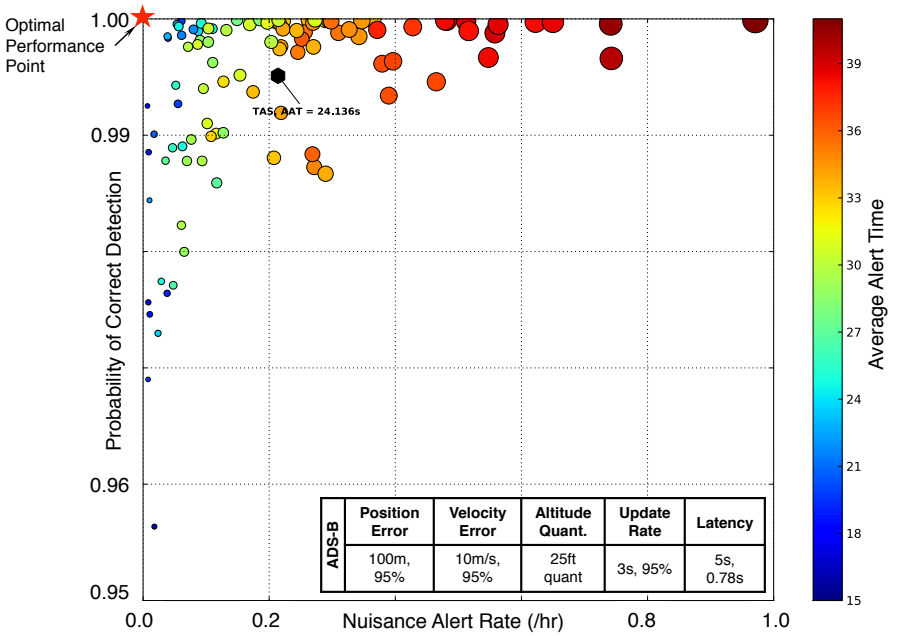


Figure 6-2: TSAA Sample Algorithm Performance of 100 Hypercube Points for the ADS-B Nominal Target

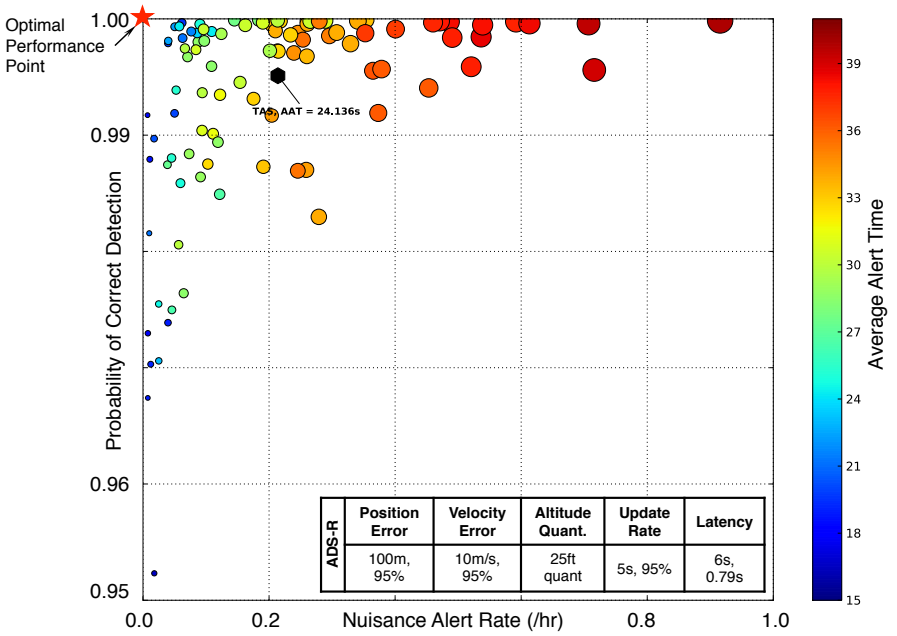


Figure 6-3: TSAA Sample Algorithm Performance of 100 Hypercube Points for the ADS-R Nominal Target

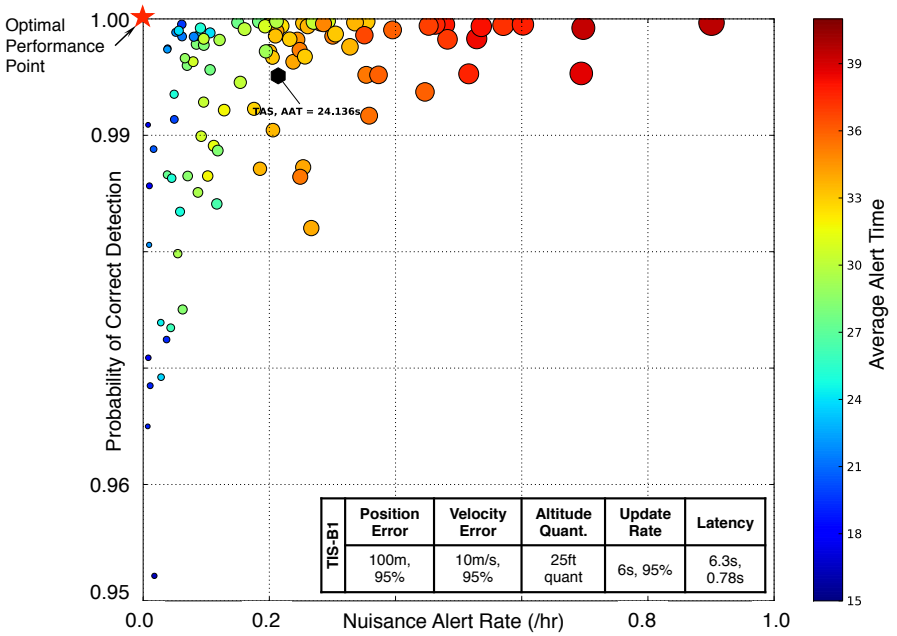


Figure 6-4: TSAA Sample Algorithm Performance of 100 Hypercube Points for the TIS-B1 Nominal Target

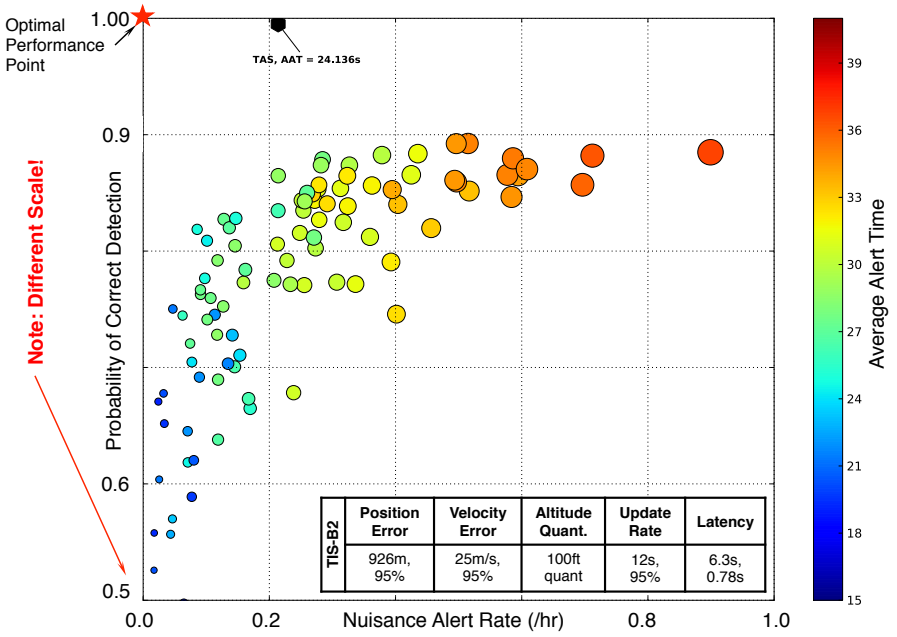


Figure 6-5: TSAA Sample Algorithm Performance of 100 Hypercube Points for the TIS-B2 Nominal Target

TSAA's performance across the nominal targets is consistent, with the exception of the TIS-B2 target. A slight shift to the left and down can be observed as the level of uncertainty increases. However, between the TIS-B1 and TIS-B2 targets, the performance significantly degrades, mainly in the late and missed alert percentage performance metrics. Based simply on the observations from these plots, it is apparent that at some point between the nominal TIS-B1 and TIS-B2 target the level of uncertainty increases such that the performance of the TSAA algorithm is significantly reduced. Other than the TIS-B2 target, a significant number of algorithm combinations exceed the performance of the black hexagon (TCAS I performance), indicating that the TSAA algorithm, if configured correctly, has the potential to significantly out-perform the current industry standard.

Based on this performance data, a response model was fit to the 100 hypercube points of each nominal target using the HDMR approach described in section 5.6.2. Table 6-4 shows the R^2 value for the performance metrics, averaged across the nominal targets. As will be discussed in depth later, all performance metrics except the late alert percentage show a good fit of the model.

Table 6-4: R^2 Values for the TSAA Performance Metrics

Performance Metric	Model Fit (R^2 Value)
Nuisance Alert Rate	91.7%
Missed Alert Percentage	82.9%
Late Alert Percentage	19.1%
Average Time of Alert Before CPA	98.5%
Total Alert Count	92.8%

6.2.2 IDENTIFICATION OF HIGH IMPACT ALGORITHM PARAMETERS

As informed by the HDMR model, Figure 6-6 and Figure 6-7 show the performance metrics' sensitivity to the high-impact parameters for all four nominal targets. This sensitivity value can be interpreted as the percentage of variability in the technical performance measure that can be explained by a given parameter's variation, as estimated by the HDMR model. Only parameters with more than 5% influence are shown. Though the HDMR model approach does evaluate whether second or third order interactions contribute to the performance variability, none were found that contribute more than 5% of variability:

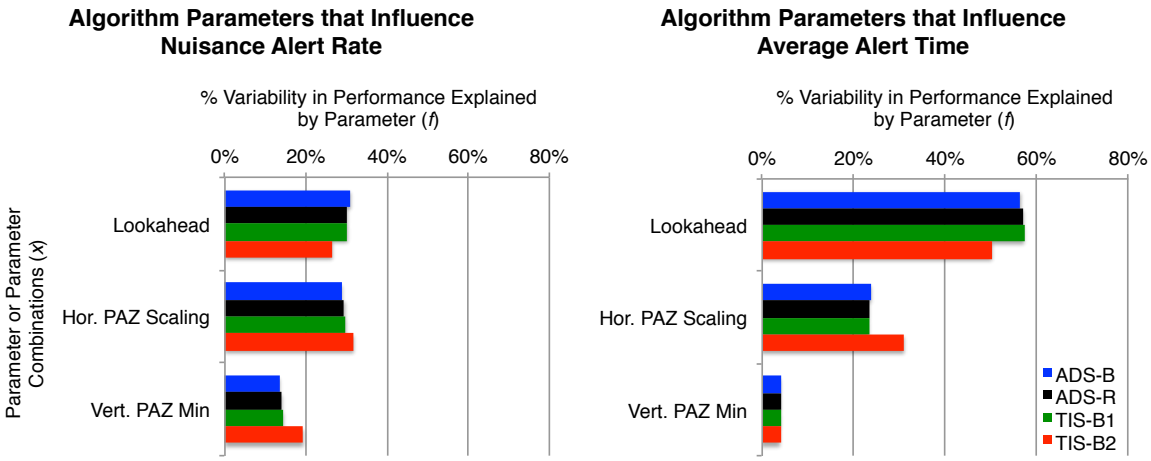


Figure 6-6: High Impact Parameters for Nuisance Rate and Average Alert Time

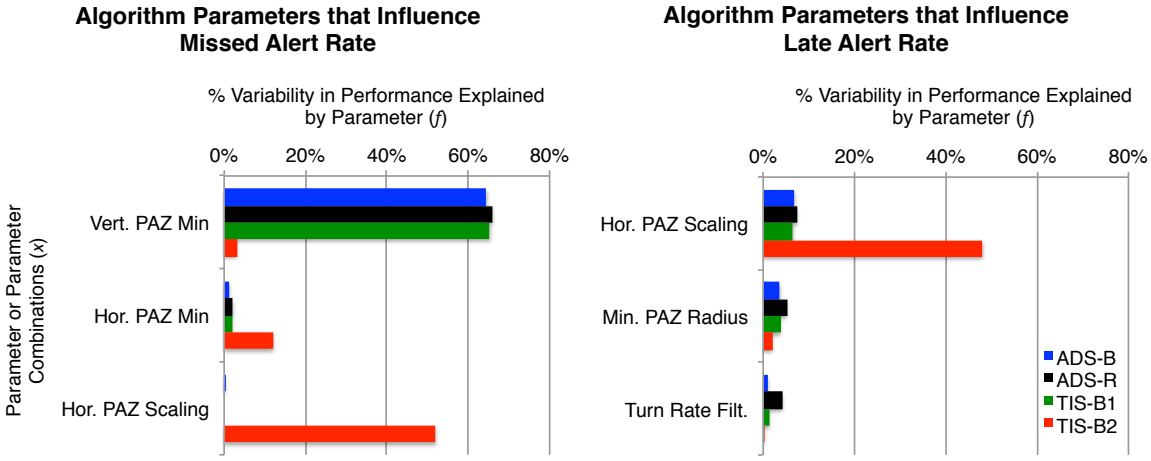


Figure 6-7: High Impact Parameters for Missed and Late Alert Percentage

As these plots make apparent, three algorithm parameters significantly affect TSAA performance: the look-ahead time (*tlook*), the factor with which the PAZ scales horizontally (*tauPAZr*) and the minimum vertical PAZ size (*vPAZmin*). Table 6-5 shows the impact of those high-impact parameters averaged across all nominal targets.

Table 6-5: Percent Variability in Performance Metrics vs. High Impact Algorithm Parameters (Averaged Across All Nominal Targets)

Parameter/Technical Performance Measure	Nuisance Rate	Correct Detections	Late Alerts	Missed Alerts	Average Alert time
Vertical PAZ Minimum	15.3%	1.1%	1.6%	49.7%	4.2%
Look-ahead Time	29.3%	2.9%	3.8%	2.4%	55.2%
Horizontal PAZ Scaling	29.9%	17.9%	17.2%	13.0%	25.4%
Total:	74.5%	21.9%	22.6%	65.1%	84.8%

Table 6-5 also shows the sum of the three main contribution parameters. This sum identifies how much of the variability in the performance metric can be explained by just the sum of those three high-impact parameters. Again, the late alert percentage (and by extension the percent correct detection) does not show good predictability by the HDMR model.

However, the average impact across all types of targets is not the whole picture. As Figure 6-7 demonstrates, as the amount of error in the state data increases, the effect of certain parameters on a particular performance metric becomes more significant. This is especially apparent in the relationship between the missed alert percentage and the PAZ scaling factor: for the target with the highest state uncertainty (TIS-B2 nominal target), the PAZ scaling factor influences the late and missed alert percentage significantly more than targets with better data quality. As later sections will discuss in detail, this is due to the fact that a larger PAZ is beneficial in the presence of large horizontal position errors, which is common with the TIS-B2 nominal target. For the ADS-B, ADS-R, and the TIS-B1 targets, however, the effect of a particular algorithm parameter on the performance metrics is generally within 3% of each other.

6.2.3 EVALUATING PARAMETER TRADE-OFFS FOR HIGH-IMPACT PARAMETERS

Each of the three high-impact parameters in Table 6-5 affects each performance metric differently. In general, however, there is a clear trade-off between two performance metrics: in the case of the look-ahead time, for example, there is a strong trade-off between the nuisance alert rate and the average alert time. The trade–offs for all three parameters that have the most significant impact on TSAA were evaluated individually. Given the similarity of the parameter impact across targets, the visualizations are shown for the ADS-B nominal target data only unless noted otherwise.

Weak Trade-Off Between Missed Alert Percentage and Nuisance Alert Rate Via Minimum Vertical PAZ Size

The minimum PAZ size selection has a significant impact on the missed alert percentage and a weak impact on the nuisance alert rate. Figure 6-8 shows the relationship between the missed alert percentage and the minimal vertical PAZ size. There is a very distinct improvement in the missed alert percentage once the minimum vertical PAZ dimension increases to above 350 ft. Increasing the vertical dimension to above 450 ft confers additional benefit. On the contrary, though there is a generally positive relationship between the minimum PAZ height and the nuisance alert rate, no clear trend can be distinguished. Nonetheless, the trend indicates that a lower vertical PAZ minimum would provide a better nuisance alert rate.

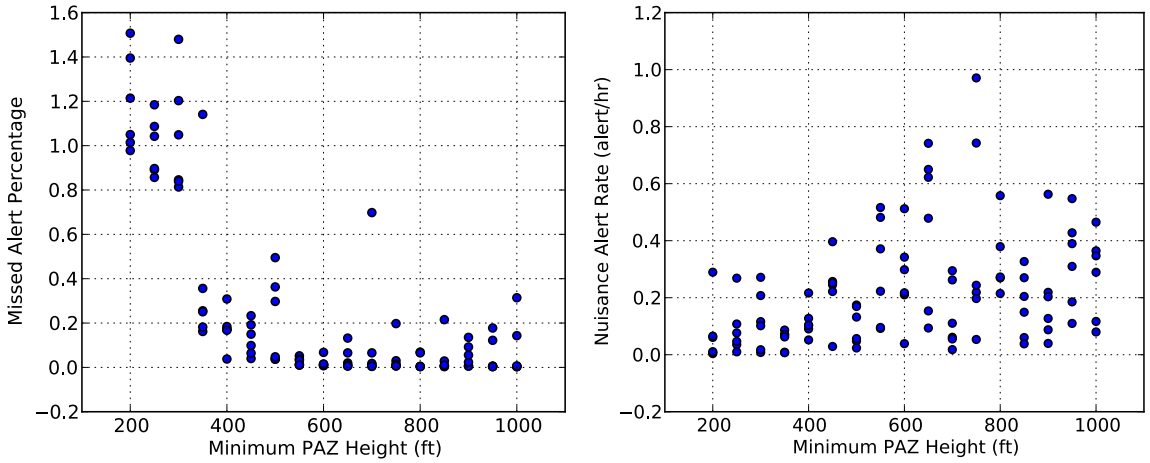


Figure 6-8: Missed Alert Percentage and Nuisance Alert Rate vs. Minimum PAZ Height (ft)

Given the data in Figure 6-8, the vertical minimum PAZ size selected was 450 ft. This selection is also beneficial given the fact that aircraft are frequently separated 500 ft vertically; thus, an aircraft passing overhead at 500 ft with no vertical trend will not generate and alert.

Trade-Off Between Nuisance Alert Rate and Average Time of Alert Before CPA Via Look-Ahead Time

The look-ahead time most significantly influences the nuisance alert rate and the average time of alert before CPA. This trade-off is visualized in Figure 6-10. As before, each dot represents one of the 100 algorithm parameter combinations and the size of the dot represents the total number of alerts issued. However, the color here represents the value of the look-ahead time used in the particular parameter combination.

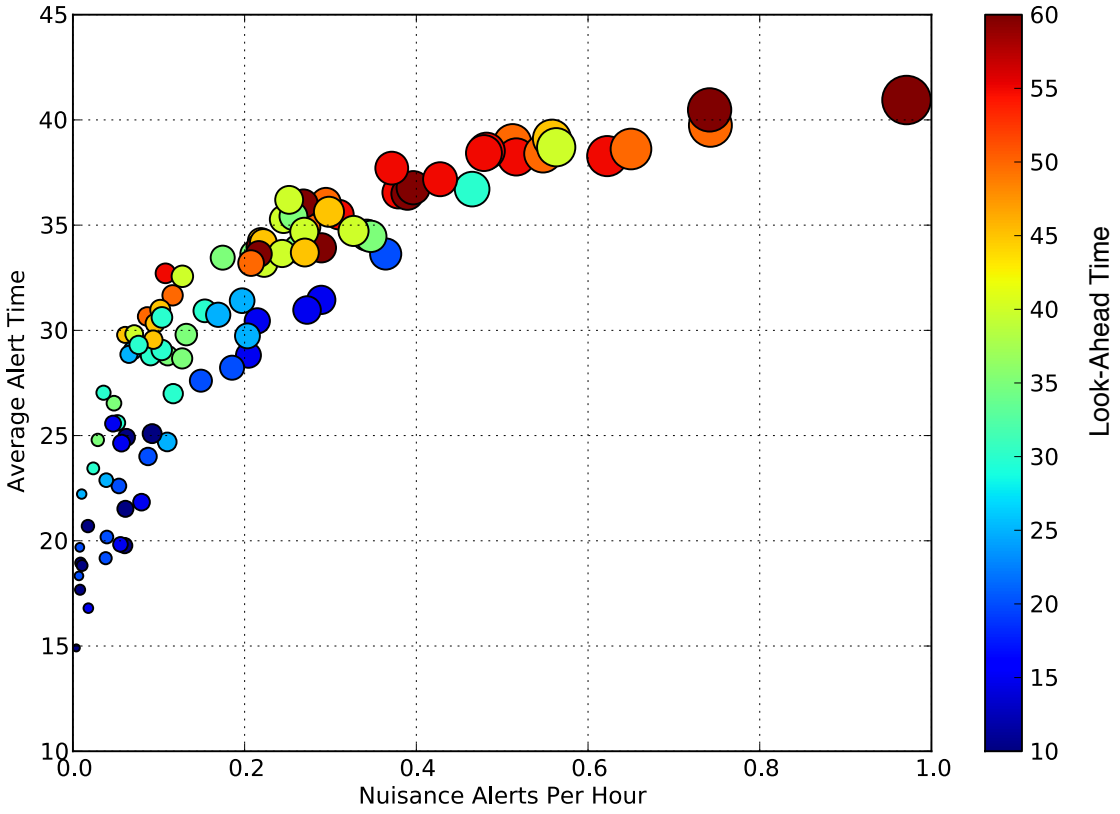


Figure 6-9: Nuisance Alert Rate vs. Average Alert Time Trade-Off for Look-Ahead Time

This figure reveals two general trends. First, as look-ahead time increases, both the average alert time and the nuisance alerts increase. Initially, a significant improvement in average alert time can be achieved at a low nuisance alert cost. As the look-ahead time increases further, however, the additional time of alert before CPA comes at a more and more significant cost in terms of nuisance alerts. Second, a significant amount of the algorithm parameter combinations fall toward the left side of the plot, which indicates that higher levels of nuisance alerts are generated by a small set of very poor combinations.

Figure 6-10 is a plot of how the individual performance metrics trade against the look-ahead time. As would be expected from the values shown in Table 6-5, the trend of the average time of alert before CPA is much clearer than the trend in the nuisance alert rate (55% vs. 29%). Also, at 35 seconds, the nuisance alert rates start to increase more significantly and the average alert time begins to level off. It is important to note that Figure 6-10 is a projection of the five-dimensional performance space onto a two-dimensional plot, resulting in increased variability in the y-direction for a given x-value.

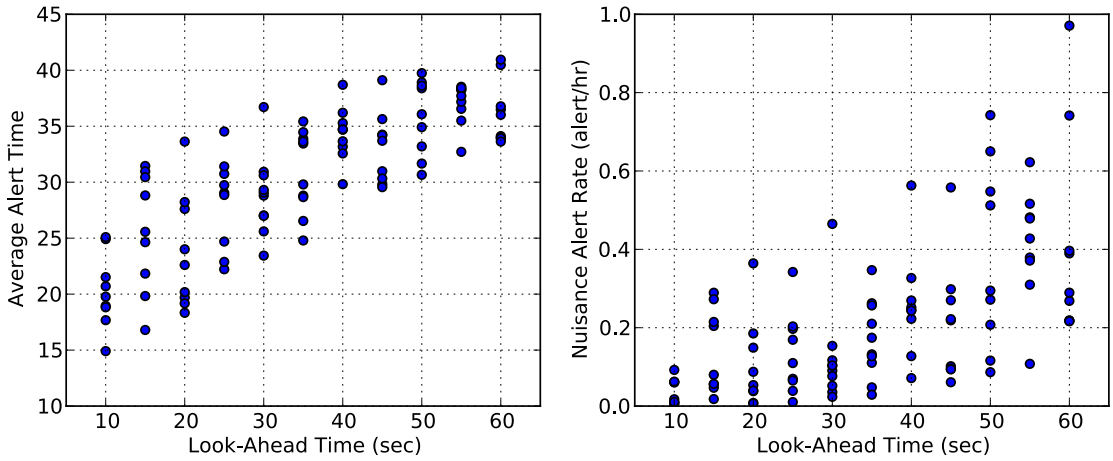


Figure 6-10: Look-Ahead Time Trade-Off Between Nuisance Alert Rate and Average Alert Time For the ADS-B Nominal Target

The look-ahead time parameter has the most significant effect on the average time before CPA that an alert is issued. As section 3.4 discussed in depth, maximizing the average time of alert before the closest point of approach is important to prevent aggravating encounters with aircraft equipped with avoidance systems. In combination with the setting of the horizontal PAZ scaling factor discussed next, setting the look-ahead time to 35 seconds maximizes the average alert time while maintaining reasonable nuisance alert rates.

TRADE-OFF BETWEEN NUISANCE ALERT RATE AND AVERAGE TIME OF ALERT BEFORE CPA VIA THE HORIZONTAL PAZ SCALING FACTOR

Similar to the look-ahead time, the horizontal PAZ scaling factor parameter has the most significant influence on both the nuisance alert rate and the average time of alert before CPA. Compared to the look-ahead time, however, the scaling factor affects the nuisance alert rate more significantly than the average alert time. Also, the horizontal PAZ scaling factor has at least a moderate effect on all performance metrics (~10%) as Table 6-5 demonstrates. The nuisance alert rate vs. average alert time trade-off via the horizontal PAZ scaling factor is visualized in Figure 6-11. Again, each dot represents one of the 100 algorithm parameter combinations and the size of the dot represents the total number of alerts issued. However, the color here represents the value of the horizontal PAZ scaling factor used in the particular parameter combination.

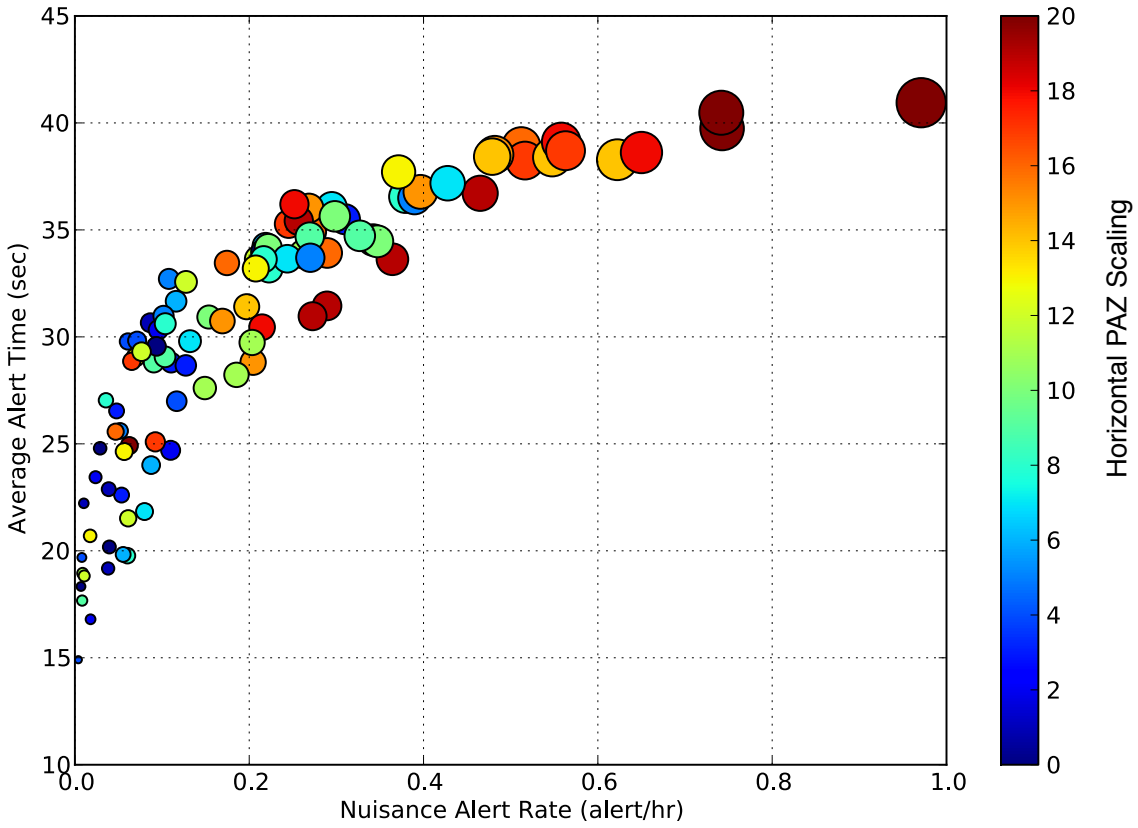


Figure 6-11: Nuisance Alert Rate vs. Average Alert Time Trade-Off for PAZ Scaling Factor

When compared to Figure 6-9, the shape of the trade-off between the average alert time and the nuisance alert rate in Figure 6-11 is identical in shape but not coloring. In general, higher horizontal PAZ scaling values result in higher average alert times. It stands to reason that the scaling factor would have a similar impact on the performance metrics as the look-ahead time as it effectively (albeit indirectly) increases the total look-ahead time. However, the effect of the horizontal PAZ scaling factor on the two performance metrics is much less consistent than that of the look-ahead time, as is evident by how much more mixed the colors are throughout the plot. This effect is also very noticeable when the scaling factor is plotted against the individual performance metrics, as shown in Figure 6-12.

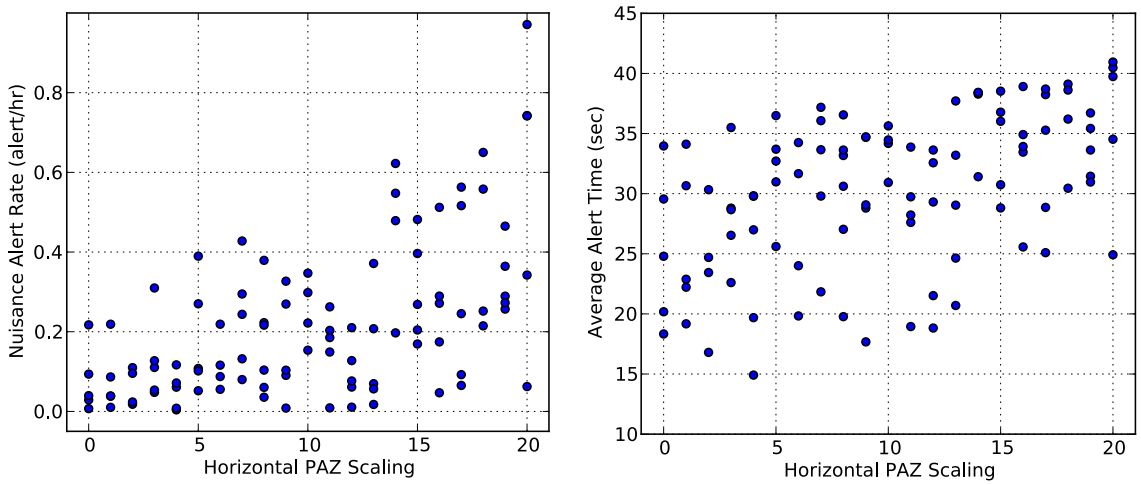


Figure 6-12: Horizontal PAZ Scaling Trade-Off Between Nuisance Alert Rate and Average Alert Time For the ADS-B Nominal Target

It would be a challenge to select the value for the horizontal PAZ scaling factor based on the plots in Figure 6-12. Though a larger scaling factor appears to result in generally higher nuisance alert rates, low nuisance alert rates are possible at higher scaling factors as well.

Given the limited usability of the trade-off visualizations from the nominal ADS-B target data, the plots in Figure 6-6 and Figure 6-7 were evaluated again. The sensitivity of the nuisance alert rate and average alert to the horizontal PAZ scaling factor is similar across all targets. However, evaluating the percentages of late and missed alerts shows that their sensitivity to the horizontal PAZ scaling factor is significantly larger for the TIS-B2 nominal target than any other target. Table 6-6 highlights how this difference is very apparent when the effect of the horizontal PAZ scaling factor for the TIS-B2 target only (not averaged across all targets) is evaluated.

Table 6-6: Percent Variability in Performance Metric vs. Horizontal PAZ Scaling for the TIS-B2 nominal Target

Parameter/Technical Performance Measure	Nuisance Rate	Correct Detections	Late Alerts	Missed Alerts	Average Alert time
Horizontal PAZ Scaling	31.70%	53.92%	47.96%	51.89%	31.10%

Given this observation, it is clear that the horizontal PAZ scaling factor has the potential to improve missed and late alert percentages for low quality targets.[16] The dominant trade-off for the horizontal PAZ scaling factor is no longer between the average alert time and nuisance alert rate, but rather between the percent correct detection and the nuisance alert rate. Figure 6-13 shows this trade-off for the TIS-B2 nominal target performance data. As is quickly evident, a larger horizontal PAZ scaling factor contributes to an increased probability of correct detection for a target associated with higher levels of state uncertainty.

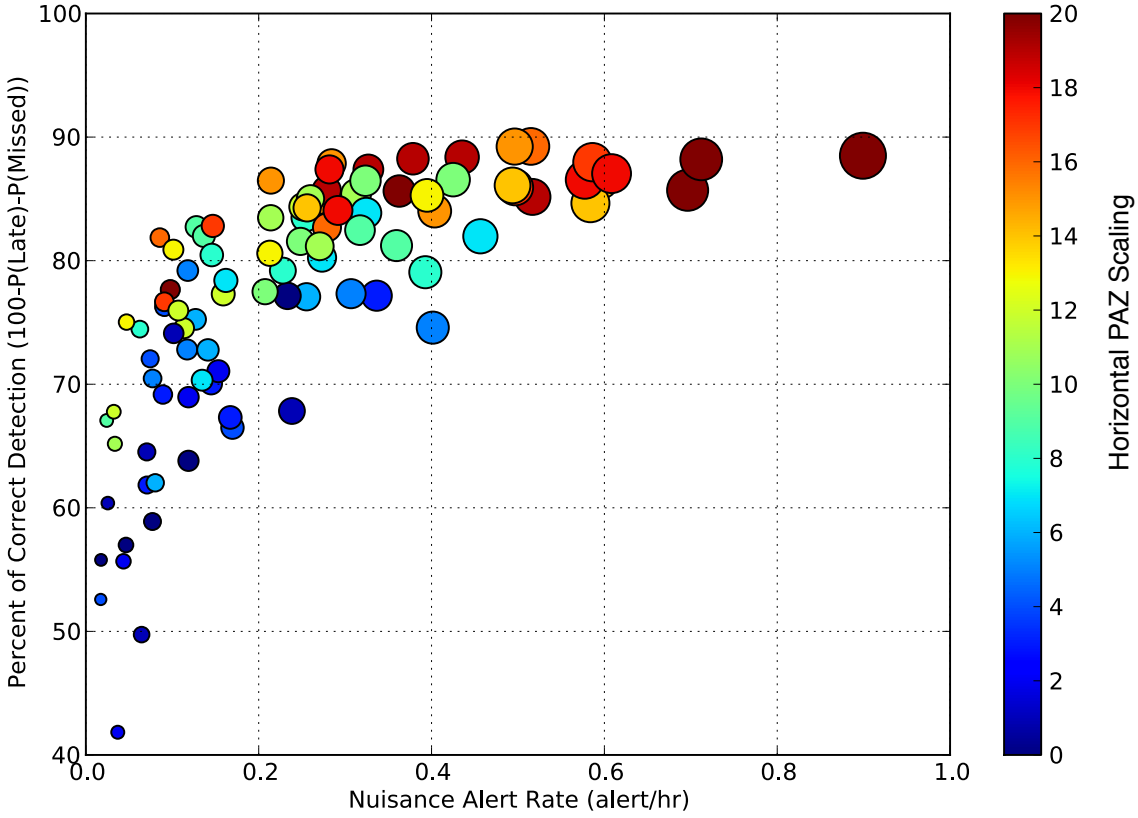


Figure 6-13: Percent Correct Detection vs. Nuisance Alert Rate Trade-Off via Horizontal PAZ Scaling for the Nominal TIS-B2 Target

[16] This observation is enabled in particular by employing this approach to tuning the TSAA algorithm. It would be challenging to identify this type of interaction using a utility function approach unless this particular trade had been included in the utility function.

Figure 6-14 shows the trades of the horizontal PAZ scaling factor against the individual performance metrics, again for TIS-B2 nominal target performance data.

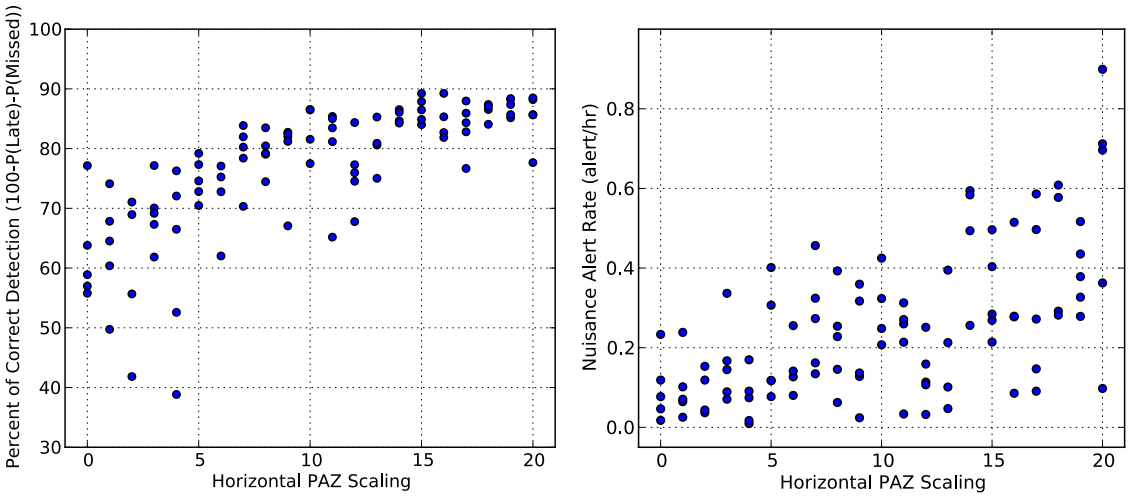


Figure 6-14: Percent Correct Detection and Nuisance Alert Rate vs. Horizontal PAZ Scaling for the TIS-B2 Nominal Target

As Figure 6-14 shows, there is a stronger positive relationship between the horizontal PAZ scaling factor and the probability of correct detection. At the same time, though not as clear, there is a positive relationship between the scaling factor and the nuisance alert rate.

Before the final value of the PAZ scaling factor can be selected, a few additional observations must be made. First, the TIS-B2 target was deliberately modeled as a worst-case target with a high 95% bound on the position error and a low update rate. As such, the TIS-B2 target represents a corner case and therefore should not be used as the driving case of the TSAA algorithm design. Second, though TSAA as a system is required to operate on TIS-B targets, TIS-B targets are expected to become a smaller and smaller percentage of the targets encountered by a TSAA system as NAS-wide as ADS-B equipage increases and the number of TIS-B targets decreases. Third, TSAA is primarily a system for ADS-B targets. Therefore, TIS-B targets should not drive the tuning of the overall TSAA system performance. Finally, though the horizontal PAZ scaling factor can improve correct detections in the presence of higher levels of state uncertainty, it also can increase the nuisance alert rate, as plots above show. Given these considerations, the final value selected for the horizontal PAZ scaling factor was 8 seconds.

An additional but secondary consideration for the selection of 8 seconds was that combining it with the selection of 35 seconds of look-ahead time resulted in an effective

look-ahead time of 43 seconds. Per Table 3-7, this allows alerts to be issued ahead of the time required to prevent undesirable interactions with avoidance systems up to 20,000 ft.

6.2.4 TSAA v5 Parameter Selection

The process described above was repeated for all other parameters and all nominal targets in order to identify whether there were any natural performance break points. However, as mentioned above, the remaining five parameters had a very limited effect on the overall performance metrics. As a result, the selection of the settings for the remaining parameters was informed by non-performance system requirements and additional considerations such as operational insights or engineering limitations. Table 6-7 lists the final parameter settings and the logic used to arrive at each one of them is identified.

Table 6-7: Final TSAA v5 Parameters and Reasoning for Selection

Algorithm Internal Parameter	TSAA v5 Parameter	Optimized Trade-Off or Selection Logic
Look-ahead Time (s)	35	Maximize average alert time while minimizing nuisance alert rate
Trajectory Discretization (s)	1	Limited impact on performance, reduce computational load
Turn Rate Filter (#)	5 updates	Very weak trade between nuisance alert rates and late alerts
Vertical Rate Filter (#)	11 updates	Values below 11 result in oscillatory behavior in vertical rate estimation and values above 11 result in sluggish response
Min. PAZ Radius (ft)	750	Very weak trade between nuisance and late alerts, selected based on common pattern separation distances during high-density operations
Min. PAZ Height (ft)	450	Minimize missed and late alerts for targets with lower levels of state uncertainty
Hor. PAZ Scaling (s)	8	Minimize missed and late alerts for targets with high levels of state uncertainty; increases look-ahead time to where interactions with avoidance systems are potentially minimized
Vert. PAZ Scaling (s)	2	Overall very weak effect on performance metrics; increasing effect on nuisance alerts at higher levels
Conflict Search Frequency (s)	Once per second	Selected to match the update rate of the DO-317A tracker
Target Discontinuation Threshold (s)	15	Selected to discontinue targets in view of a single en-route radar after one missed update

CAZ Radius (ft)	500	Selected based on maximum allowable position uncertainty of two rule compliant ADS-B targets
CAZ Height (ft)	200	Selected based on 100ft altitude quantization common in General Aviation aircraft
Re-alert Delay (s)	6	Selected based on maximum expected duration of the aural alert issued by TSAA
Double Trigger	15	Selected based on the 12.5 reaction time requirement identified by [90]

The exemplar algorithm is optimized in a single configuration for all types of targets. More consideration was given to high-quality ADS-B targets in the situations where the optimal parameter selection did not agree across all nominal targets. Most commonly, the optimal parameters agreed between the ADS-B, ADS-R, and TIS-B1 target and differed slightly for the high state uncertainty TIS-B2 target.

However, some of the insights gained from the tuning process could potentially inform different algorithmic approaches, two of which are discussed here.

First, instead of implementing TSAA with a single parameter combination, two or more separate parameter combinations could be implemented. Given a particular target, the most appropriate of the parameter combinations would be selected. For example, one parameter combination could be used for targets like the nominal ADS-B, ADS-R and TIS-B1 target, and another for TIS-B2-like targets.

A second option would be to make the parameters that most directly protect against uncertainty dependent on the target's reported level of uncertainty. For example, the minimum PAZ size could be adjusted to match the reported level of position uncertainty, which effectively would extend the concept that originally sized the CAZ to non-rule compliant ADS-B targets. Alternatively, targets with higher levels of position uncertainty could be assigned higher horizontal PAZ scaling factor. It is expected that once the exemplar algorithm is published in the standard, manufacturers soon will identify other algorithmic approaches as well.

6.3 Analysis of TSAA Algorithm Performance with Tuned Parameters

Given the tuned parameters settings from the previous section, the sample TSAA algorithm's performance was evaluated in-depth. A summary of the results is presented here.

6.3.1 Comparison of TSAA Performance to TCAS I (TAS) Performance

In order to compare the performance of TSAA to the current industry standard, the same data set was evaluated using a basic implementation of the TCAS I algorithm. The algorithm was written to meet DO-197A, the TCAS I Minimal Operational Performance Standard (MOPS). It should be noted that only the conflict alerting algorithm was implemented for this analysis; in real implementations, additional trackers and manufacturer-introduced heuristics likely would improve performance from the base achieved with this basic implementation.

Using the standard range error model described in the TCAS II standard (DO-185), the range error was modeled as a constant bias sampled from a uniform distribution over -125 to 125 ft at the beginning of each encounter. In addition, a time-changing jitter is sampled from a normal distribution with 50 ft sigma once per update and added to the range measurement once per second. The range tracker is then modeled as an alpha-beta tracker with alpha of 0.67 and a beta of 0.25 as described in DO-185. The altitude error is modeled with the same model described in section 5.5.2 and the vertical tracker is configured identically to the range tracker. Table 6-8 shows the performance results of the tuned TSAA algorithm for all four nominal targets compared to the performance of TCAS I for the same data set.

Table 6-8 also shows TSAA's performance when it is run on non-degraded truth data. From an analytical point of view, TSAA's performance on the truth data is the theoretical limit of the performance capabilities of TSAA. The data in Table 6-8 is visualized in Figure 6-15. As a reminder, smaller numbers indicate better performance.

Table 6-8: Performance of Tuned TSAA Algorithm Nominal Targets with Comparison to TCAS I

Performance Metric	TCAS I	ADS-B Target	ADS-R Target	TIS-B 1 Target	TIS-B 2 Target	No Degradation
Nuisance Alert Rate (hrs/alert)	4.68	9.65	10.41	10.42	6.86	11.823
Percent Correct Detection	99.51	99.92	99.91	99.83	80.5	99.992
Percent Late Alerts	0.428	0.044	0.051	0.12	7.9	0.03
Percent Missed Alerts	0.061	0.038	0.039	0.051	11.7	0.0
Average Alert Time (sec before CPA)	24.136	30.6	30.4	30.3	28.5	32.0
Total # of Alerts	18623	9062	8496	8508	11373	7465
Alerts on Unknown CPAs	1955	162	157	168	479	107
Suppressed Alerts (Ground, etc.)	10573	2395	2314	2276	3185	2049

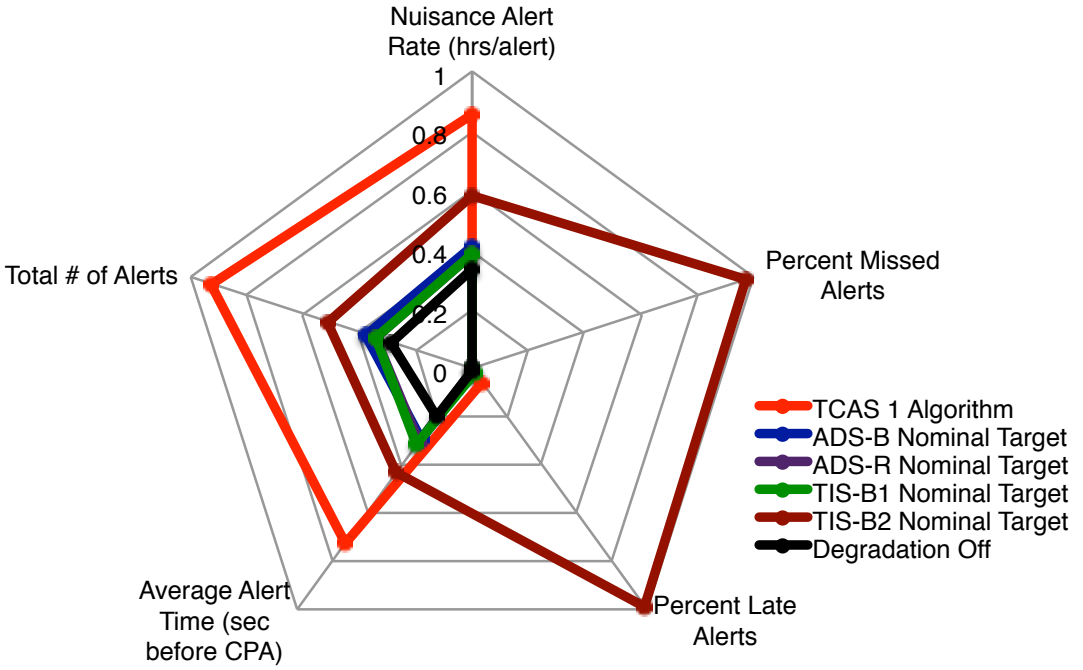


Figure 6-15: Polar Plot of TSAA Performance for all Four Nominal Targets, No Error and a Basic TCAS I Algorithm Implementation (smaller is better)

As Table 6-8 and Figure 6-15 show, the performance of TSAA is superior to the performance of the basic TCAS I implementation for targets with limited amounts of state uncertainty. TSAA issues half as many nuisance alerts, late and missed alert rates are lower, and the average alert time is 6 seconds longer compared to TCAS I. Figure 6-16 shows the location of TSAA performance with tuned parameters in relation to the subset of the 100 hypercube points for the ADS-B nominal target in the top left corner of the Figure 6-2. As can be seen, the TSAA algorithm with the tuned parameters is able to push into an area of generally cool average alert time coloring with a warmer, small dot.

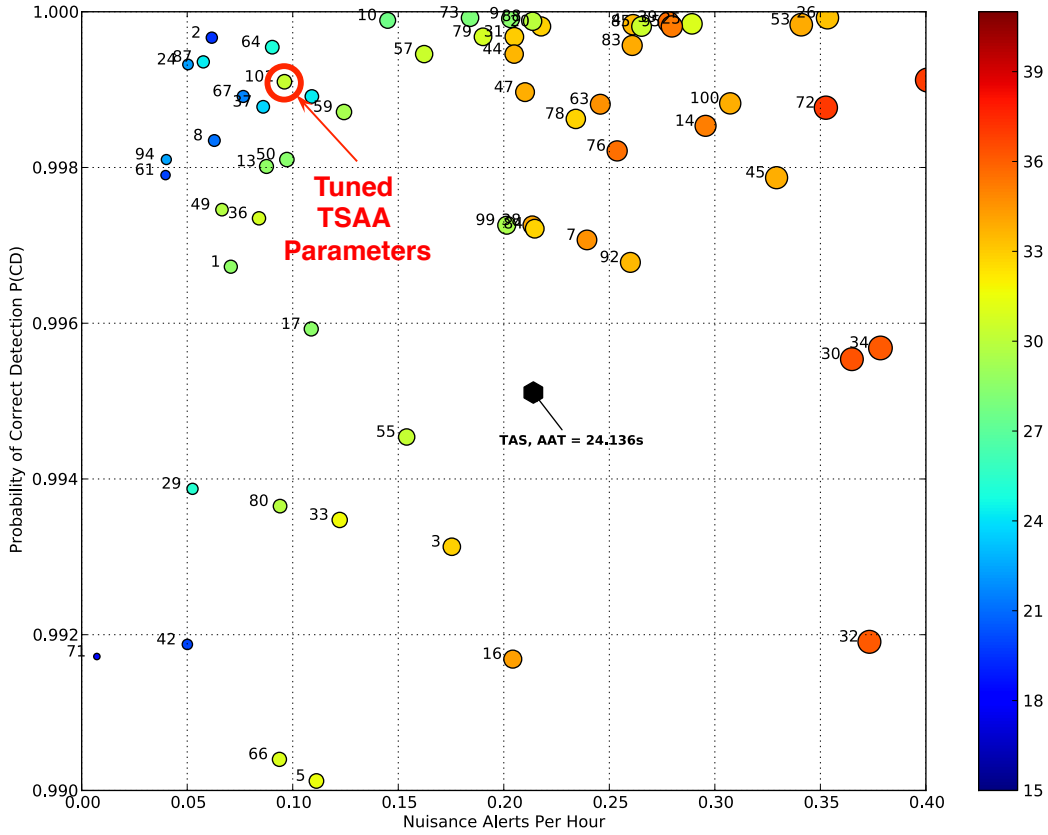


Figure 6-16: Location of TSAA Performance with Tuned Parameters in Relation to 100 Hypercube Points (ADS-B Nominal Target)

6.3.2 NUISANCE ALERTS DUE TO NOISE IN THE ESTIMATED TURN RATE ("TRAJECTORY WAGGING")

Since the exemplar algorithm uses a constant turn rate trajectory prediction, the algorithm is very sensitive to noise in the turn rate. This noise can be introduced by the flight crew via flight technical error or by the sensor measuring velocity magnitude and direction. The noise in the turn rate effectively causes the trajectory to "wag" back and forth and solicit alerts, which causes increased rates of nuisance alerts. This causes the algorithm to predict a PAZ penetration one second, but to turn the alert off the next second after the trajectory wagged the other direction, which means the penetration is no longer predicted. In certain cases, this occurred multiple times during a single encounter, causing frequent re-alerts over its duration. When evaluating the duration of TSAA alert, this becomes very apparent. Figure 6-17 shows a histogram of how long a given alert remains "on" after being issued by TSAA (i.e., alert duration). Note that six seconds is the minimum allowable alert duration as defined by the re-alert delay algorithm parameters that maintain a given alert and prevent any other alerts from being issued while the initial aural alert is being pronounced.

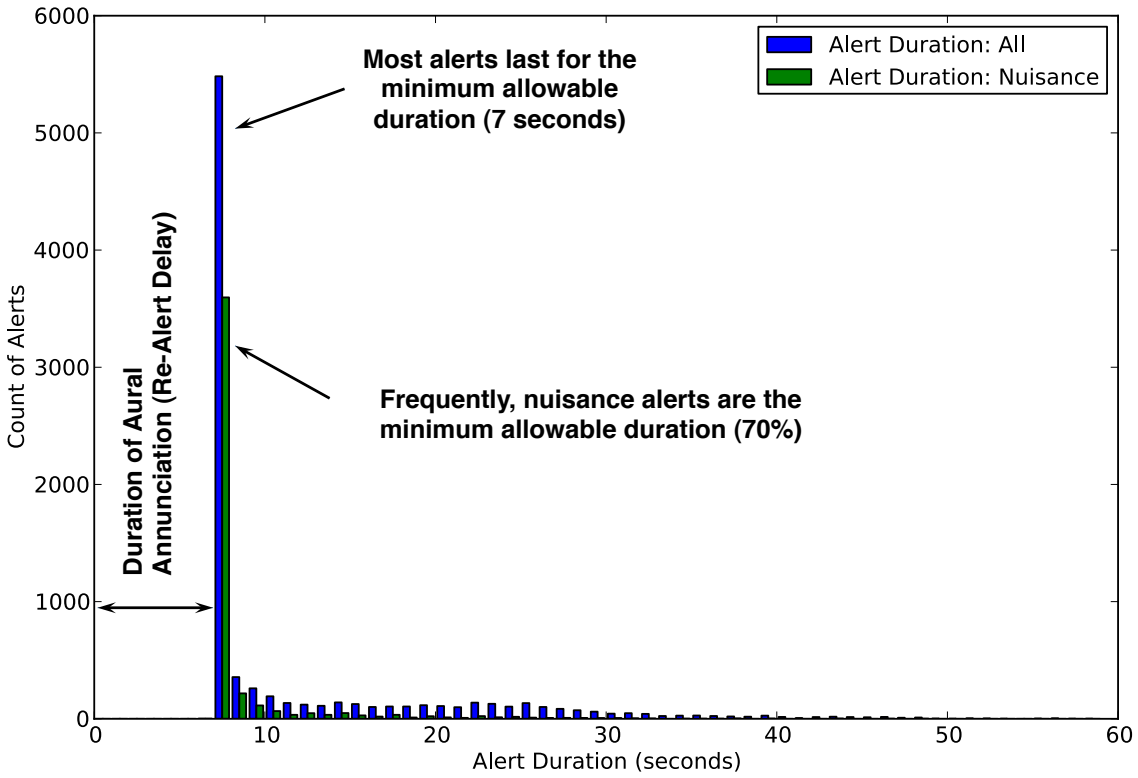


Figure 6-17: Histogram of the Duration of All Alerts (Blue) and Nuisance Alerts (Green)

As can be seen, a significant portion of nuisance alerts were this type of alert. Additionally, repeated alerts that would not be scored as nuisances objectively may be perceived as such if frequent re-alerting occurs during a single encounter. This issue was originally observed during simulations but was worse during flight test. Most notably, this type of alert occurred frequently during operations in the airport pattern when other aircraft were closest. A solution to this particular problem is implemented in the next chapter as part of the algorithm validation.

6.3.3 IDENTIFICATION OF CAUSES FOR MISSED AND LATE ALERTS

The TSAA algorithm with the tuned v5 parameters and no degradation did not miss any alerts and had only four late alerts. Upon closer evaluation, the encounters with late alerts occurred during an airshow in which two aircraft were operating 1000 ft apart and, presumably as part of the show, suddenly dove into each other, missing by a few hundred feet. In those cases, TSAA alerted immediately when the maneuver commenced but since the maneuver occurred less than 12.5 seconds before the CPA, the alert was scored as late.

However, evaluating TSAA's present errors reveals that the algorithm missed alerts more frequently and also issued alerts late. In order to evaluate what caused the algorithm to do so, a more detailed analysis was performed. Specifically, whenever an alert was missed or issued late, the position error and the altitude error was stored and post-processed. Figure 6-18 shows the distribution of the position and altitude error of the missed alerts for the nominal ADS-B target.

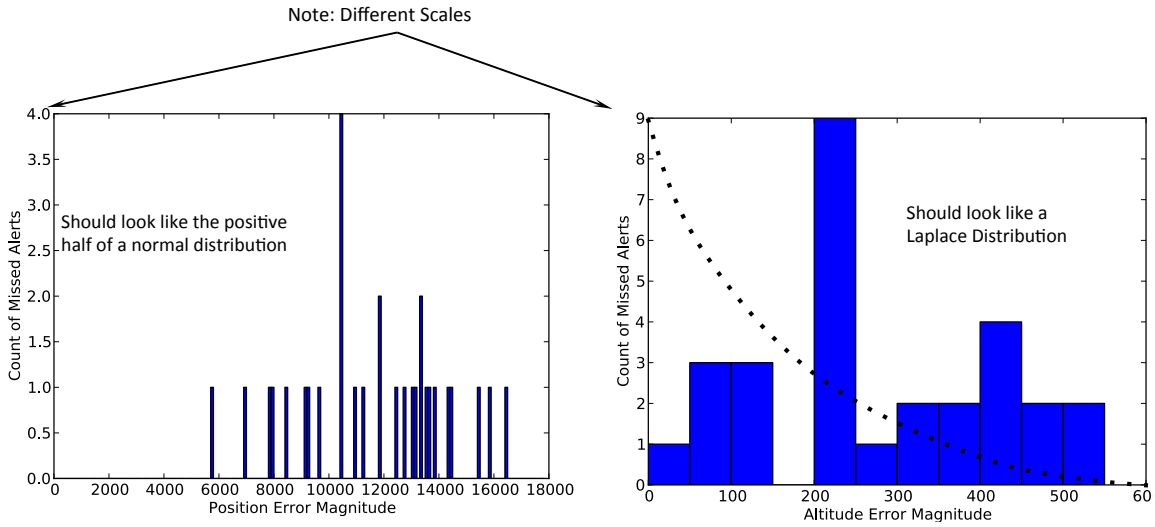


Figure 6-18: Position and Altitude Error Distribution of Missed Alerts for the ADS-B Nominal Target

Missed alerts generally were caused by large position or altitude errors. Were it another mechanism, the distribution of the errors would be expected to represent the distribution from which they were sampled initially – namely, a normal distribution and a Laplace distribution for the position and the altitude error, respectively. Missed alerts most frequently occurred in a geometry of a base/opposite base pattern encounter category of the Scripted Encounter Master Encounter Set, labeled A4b in Figure 6-19.

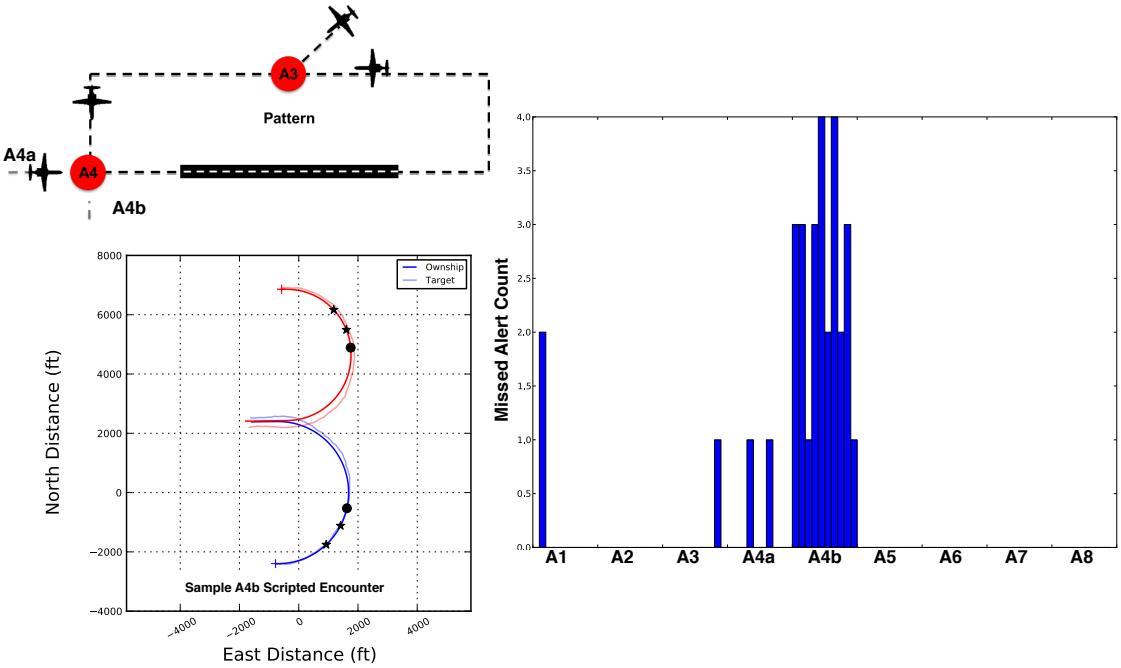


Figure 6-19: Encounters Most Frequently Missed by TSAA with Tuned Algorithm Parameters

Section 6.2.3 postulated that large position errors contributed significantly to the late and missed alerts seen for the TIS-B2 nominal target. This suspicion was confirmed when the position and altitude error analysis above was repeated for the TIS-B2 target. In Figure 6-20, the altitude error distribution represents the Laplace distribution from which the error was originally sampled. However, the horizontal position error distribution does not look like a flattened normal distribution as would be expected when sampling from a Gauss-Markovian distribution. Rather, it appears that the algorithm does not protect adequately against position errors in excess of 3,000 ft. Since the PAZ horizontal scaling factor improves the missed alert performance metric, the mechanism by which it does so must be that it protects against this higher position uncertainty. (For comparison, the TIS-B2 position error was modeled as a 3,000 ft, 95% bound error on the target plus a 300 ft, 95% bound on the own-ship.)

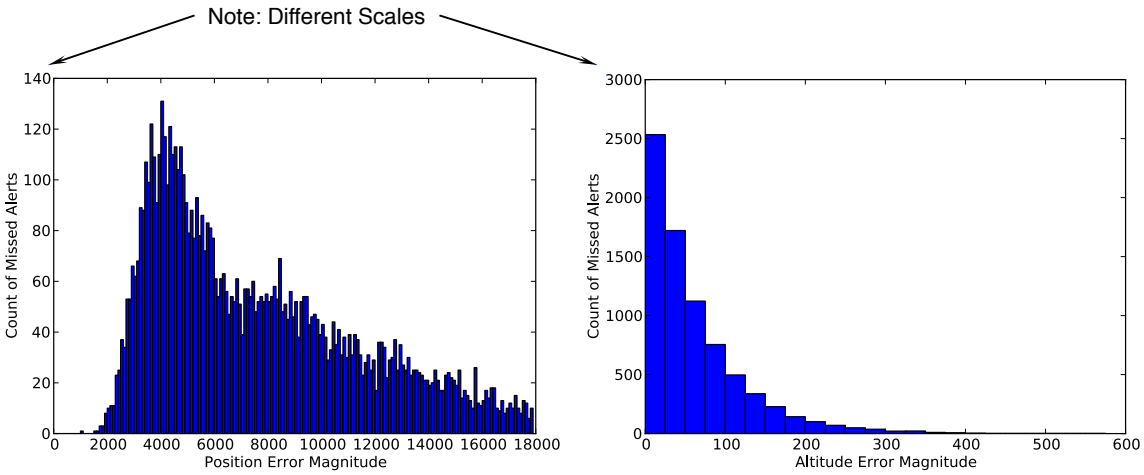


Figure 6-20: Position and Altitude Error for Missed Alerts of the TIS-B2 Nominal Target

6.4 Limitations of TSAA Performance Simulation

Throughout the process of tuning the TSAA algorithm parameters, a few lessons were learned about how the performance simulations are affected by the methods used to conduct that simulation.

The first set of lessons relates to the error models used in the degrader. One limitation the error simulation is that the position error and the velocity error were not correlated. In reality, the position error and velocity error are intimately related. Simulating them independently represents a conservative approach. A second limitation is specific to the difference in how the position error was simulated for the TCAS I algorithm implementation. For ADS-B (and TIS-B) targets, the position error was simulated as Gauss-Markovian process with a 95% bound of the NACp value. Comparatively, the range error modeled for the TCAS I algorithm was drawn from a uniform distribution of ±125 ft with an added Gaussian jitter with a sigma of 50ft as required by the standard error model. Comparatively, therefore, the errors introduced to range are bound to a smaller level than that of GNSS or radar position measurements. This is one of the sources of missed and late alerts during TSAA simulations when compared to TCAS I simulations.

The second set of lessons learned relates to the methods used to generate scripted encounters using for simulation and system evaluation. 90,000 such encounters were added to the Low Altitude and Airport Operations Master Data Set, which effectively represent 9 different scenarios in 10,000 slightly different ways to improve statistical power. Given that

the underlying scenario was identical for 10,000 encounters, any inaccuracy or limitation that was present in those 9 scenarios was effectively magnified by a factor of 10,000.

One of those effects includes the fact that encounters could be initialized in geometries would not accurately represent the dynamics of how the two aircraft arrived in the present geometry. Another limitation is the fact that the scripted encounters do not exhibit any flight technical error or operational oscillations. Such oscillations could have good or negative effects: some would increase the closure rate, causing an alert to be issued that would have otherwise been missed, while some may solicit an alert that may otherwise not have been issued.

In summary, the basic TSAA algorithm is successful at predicting airborne conflicts and provides an improvement in performance over the current industry standard. In light of this and with the satisfactory performance predicted by the simulation tool for TSAA with the tuned algorithm parameters, the next step was to evaluate TSAA during real operations. Therefore, TSAA was implemented in prototype avionics and installed in aircraft, and a comprehensive flight test program was conducted.

Chapter 7

SYSTEM EVALUATION AND FLIGHT TEST

Using the exemplar TSAA algorithm with the tuned v5 parameter settings, a comprehensive flight test program was conducted as the next step in the TSAA development effort, as shown in Figure 7-1.

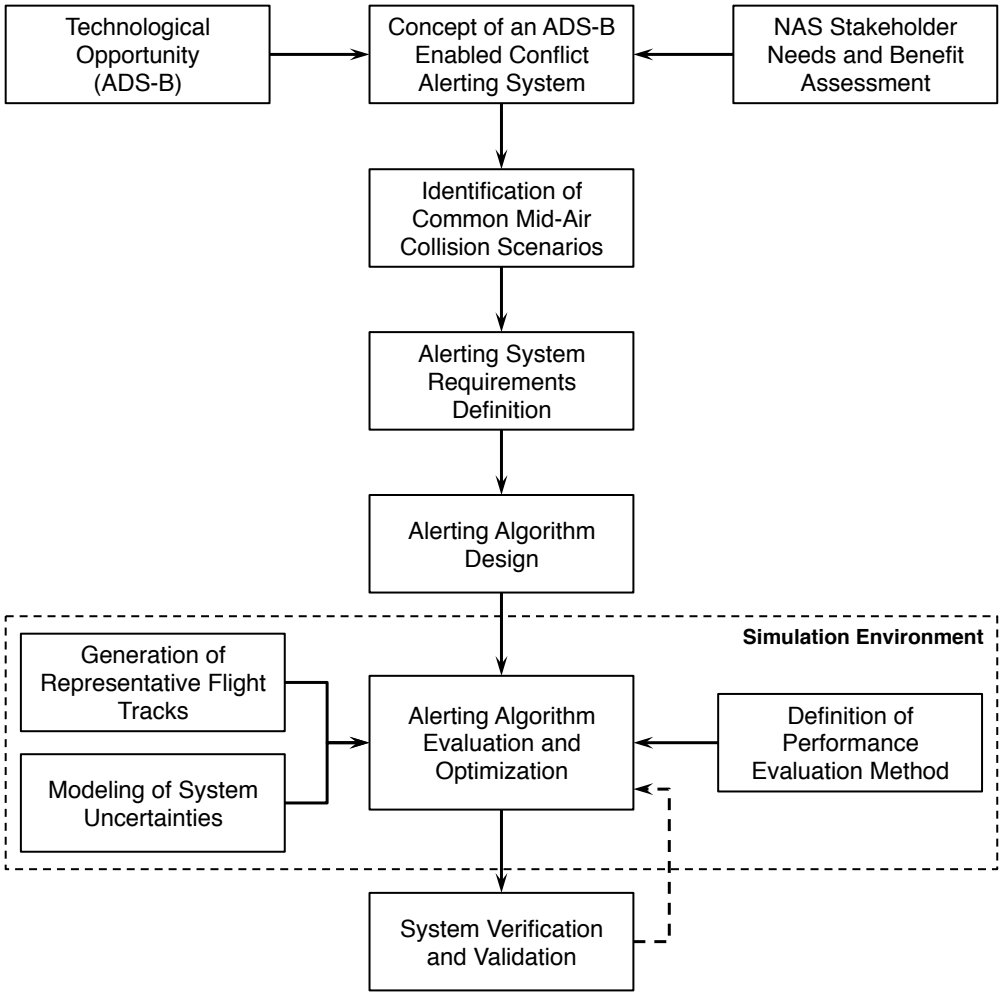


Figure 7-1: Steps Followed During the Design of TSAA

This chapter provides a high-level overview of the flight test program and presents the analysis that verified the performance of the TSAA exemplar algorithm observed during the flight tests.

7.1 Overview of the Flight Test Program

The flight test program consisted of three phases, each described at a high level here. A more in-depth discussion of the flight tests can be found in two technical reports written by the larger TSAA team. The first report, published by Avidyne, focuses on the engineering aspects and technical lessons learned during development of the prototype avionics [100]. The second report analyzed TSAA's performance from a human factors perspective [88]. Together, those reports provide a comprehensive and detailed account of the entire TSAA flight test program.

7.1.1 CHECKOUT OF PROTOTYPE AVIONICS WITH THE TSAA ALGORITHM

The early phases of the flight test focused on engineering evaluations of the hardware and whether the TSAA code was implemented correctly on that hardware. The flight tests consisted of a Cessna 182Q configured as own-ship and a Cirrus SR22 as the target and were conducted at Avidyne's Melbourne, FL flight test facility. Both aircraft were equipped with ADS-B Out and In and the prototype TSAA unit. Figure 7-2 shows a sample checkout flight. The blue line represents the own-ship track and the red line the target track. The plus sign indicates the location where the tracks originate. A black X indicates a dropout, i.e., a point when no new ADS-B update had been received for that particular target in the last 15. The yellow (PAZ) and red (CAZ) circles indicate the alert status of the target at that particular location. During this particular flight, the two aircraft flew formation with the target position at the edge of the alert region. In doing so, small changes by the target could be used to elicit an alert and evaluate whether TSAA was alerting as expected.

All in all, the checkout phase of the flight test consisted of 23 flights. Six of the flights evaluated the prototype's performance on TIS-B targets. To achieve this, the two aircraft were relocated to airspace at Marathon, FL, which was only in view of a single en-route radar (equivalent to the nominal TIS-B2 target). In order to be considered a TIS-B target, the Cirrus switched the ADS-B transponder off while the Cessna continued operating as before.

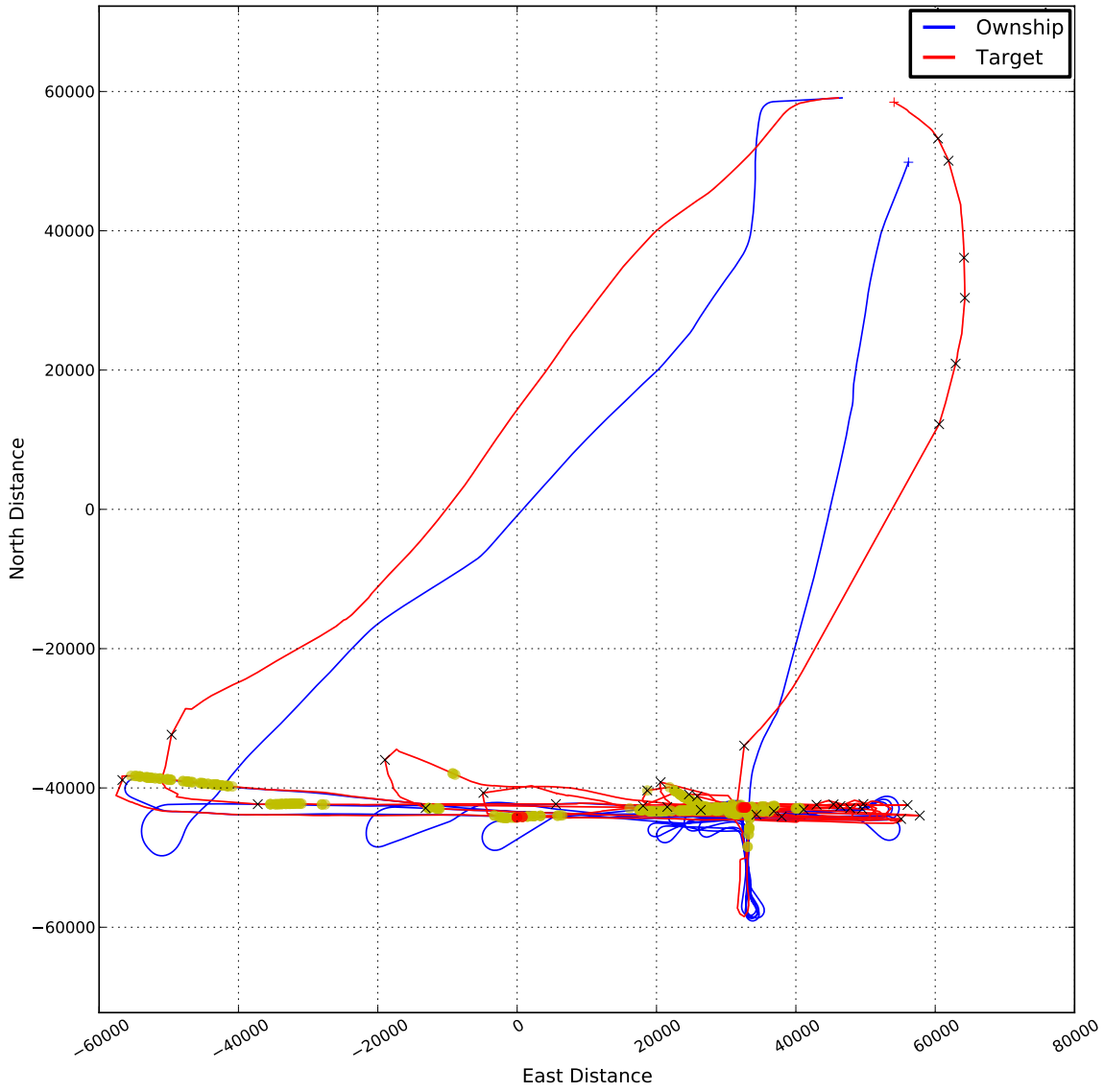


Figure 7-2: Sample Engineering Checkout Flight During Initial Flight Test Phase

Two major issues were identified and corrected during the initial evaluation of the algorithm. Initially, the prototype suffered from significant multipath interference, which caused significant dropouts. As Figure 7-2 shows, the target frequently is considered stale due to a lack of new ADS-B updates (black Xs). The second issue was a software bug that caused the prototype to call TSAA only when new data was received via ADS-B for a particular target and not once per second as originally intended. Both issues were addressed and corrected during these initial phases of the flight test program.

7.1.2 HUMAN FACTORS EVALUATIONS – SCRIPTED ENCOUNTERS AND TARGETS OF OPPORTUNITY

A group of 21 GA pilots judged the usability of the TSAA prototype from a human factors perspective during in-flight evaluations. Each participant flew the same pre-defined flight path along which 6 planned encounters occurred. Again, the two aircraft were the Cirrus and the Cessna, with the subject pilot flying the Cessna. Figure 7-3 shows a sample human factors evaluation flight (encounter IDs reference scenarios from 3.2.5).

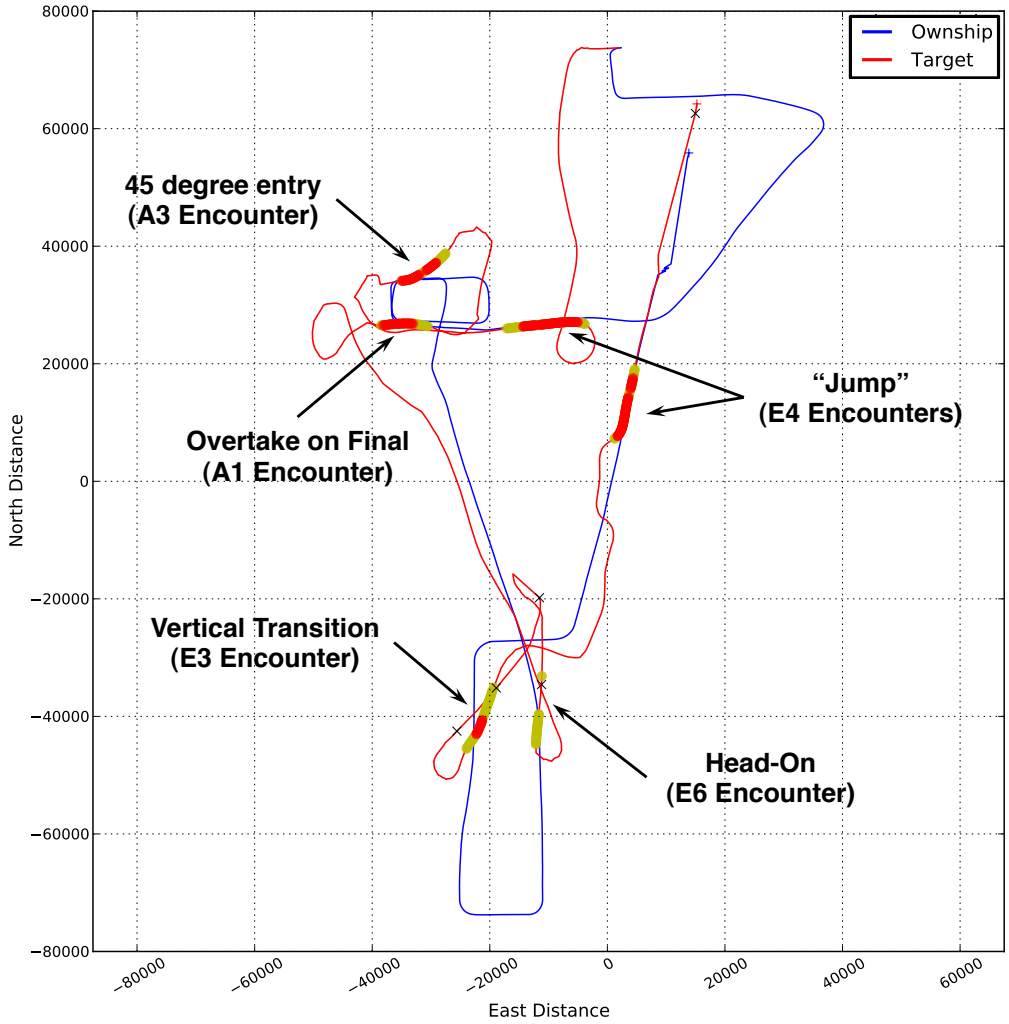


Figure 7-3: Sample Flight During Human Factors Flight Tests

Before the flight, the subject pilots were briefed about the purpose of the flights and the functionality of the TSAA. However, they were not briefed on the specific encounters that

would be presented to them during the flight and were instructed to respond as they would if they were operating normally if TSAA issued an alert. During the flight, the target aircraft positioned itself such that it could generate specific encounters as shown in Figure 7-3.

In addition to the planned encounter flights, subject pilots also were exposed to targets of opportunity by operating in the airport pattern at Daytona Beach, FL, for prolonged amounts of time. The intent of exposing the pilots to targets of opportunity in the pattern was to evaluate whether pilots perceived TSAA to be a reliable system in the airport pattern, the environment with the highest benefit potential for TSAA.

The pilots considered the alerts to be appropriate in both the planned encounters and during encounters with targets of opportunity in the airport pattern. Frequently, alerts in the pattern were considered appropriate even though the pilot did not take evasive action. One important result from the human factors flight tests was that it is imperative for TSAA to not issue alerts or traffic that is on the ground or while the own-ship is operating on the ground. Overall, the pilots reported high trust in the system after the flight and perceived it to issue alerts timely and with accurate information.

7.1.3 HIGH PERFORMANCE AND HELICOPTER TESTS AT THE FAA'S WILLIAM J. HUGHES TECHNICAL CENTER

In addition to the engineering and human factors evaluations, additional flight tests with high performance aircraft and helicopters were conducted at the FAA's Tech Center in New Jersey. These tests were conducted using the FAA's Global 500 and one of the FAA's Convair 580 aircraft. The helicopter flight test consisted of three phases. First, the FAA's Sikorsky 76 (S76) helicopter and Avidyne's Cirrus aircraft flew a set of scripted pattern encounters at the Atlantic City International Airport. Second, the S76 transitioned to the Philadelphia International Airport and established an orbit on the northwestern side of the intersection between runways 17/35 and 16/27R. The orbit was maintained for approximately one hour of flight time and any alerts issued by TSAA on arriving and departing aircraft were noted. This particular flight track is shown in Figure 7-4 with the blue line representing the track of the S76. Lastly, the S76 flew a set of encounters with a Bell 206 to generate helicopter specific encounters, which would be common during electronic new gathering or other close proximity helicopter operations. All aircraft were equipped with the Avidyne prototype TSAA unit.

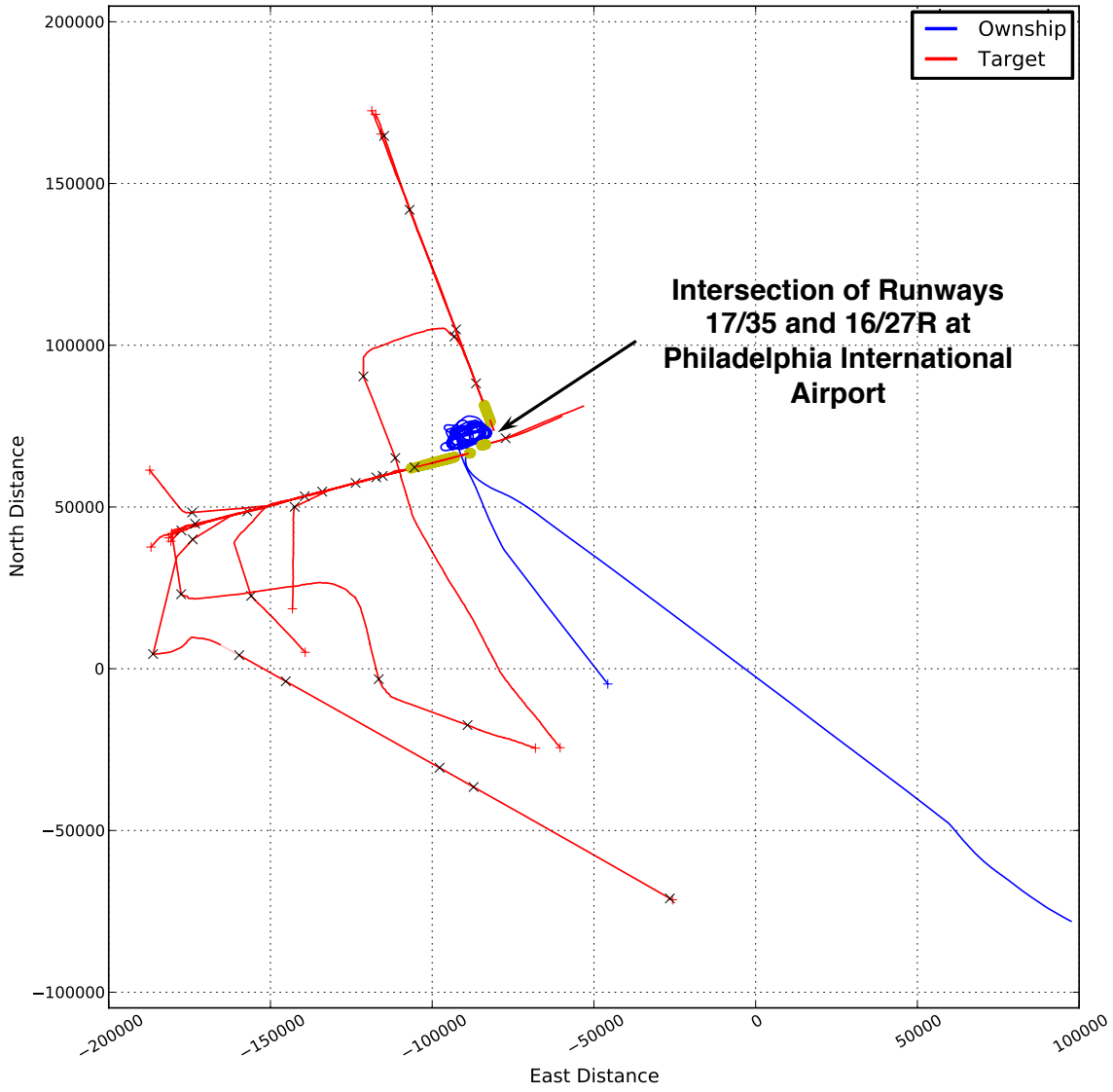


Figure 7-4: Flight Track of S76 Flight to Philadelphia International Airport

A total of 9 flights were flown for the high performance tests and 15 flights were flown for the helicopter test. Similar to the human factors evaluation flights, the test cards consisted of scripted encounters as defined by the scenarios from the mid-air collision analysis in Chapter 3; they are described in detail in Avidyne's flight test report [100].

7.1.4 FLIGHT TEST IMPLEMENTATION OF TSAA

The TSAA system was implemented differently for the flight test than it was for the performance analysis. In the prototype, the data sent to TSAA was the state data as estimated by the DO-317A tracker before it was extrapolated to a common point in time. Internal to TSAA, the constant turn rate trajectory prediction was adjusted to perform the extrapolation to a common point in time using the constant turn rate trajectory prediction. Figure 7-5 is a schematic of this implementation. However, the data sent to the display that showed the targets' location to the flight crew was implemented as required by the DO-317A standard, using a constant heading extrapolation, once per second.

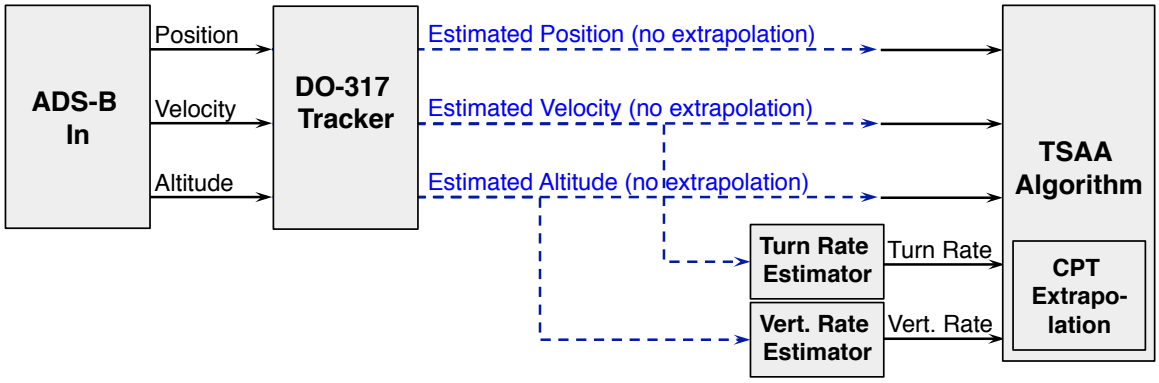


Figure 7-5: TSAA Implementation Used During Flight Test

7.1.5 LOGGING OF TSAA FLIGHT TEST DATA

The prototype TSAA system contained a logging function that logged a large amount of algorithm and overall system status data at the top of every UTC second. The logging function also saved every ADS-B message that the prototype received as well as the data that was passed on to the TSAA algorithm during flight. Lastly, it logged the time, duration, and level (CAZ vs. PAZ) of any alert issued during the flight tests. This data serves as the basis of all the analysis presented later in this chapter.

One limitation of this analysis comes from this method of logging the data. The logging function was executed at the top of every UTC second and logged the extrapolated state data that was being sent to the display (i.e., extrapolated using constant heading). TSAA, on the other hand, was called once per second but at an unknown point during that second and with the un-extrapolated state data. As a result, the logged data is not exactly the same data that that the TSAA algorithm used in the prototype unit. In light of this, when the logged data is run through the simulation tool, the alerting behavior between the two systems is not expected to be exactly the same.

7.2 Analysis of TSAA Alerting Performance during Flight Test

The analysis the TSAA algorithm's performance during the flight tests is divided into two sections. The first evaluates the aggregate alerting performance of the TSAA prototype unit ("the prototype") independent of what behavior would have been expected. The second section compares the alerting behavior of the prototype unit to the alerting behavior expected from the simulation tool.

7.2.1 Analysis Approach

Each encounter that occurred during the flight test program was analyzed individually. An encounter was defined as any time that two aircraft came closer than 5000 ft slant range or that an alert was issued by the prototype. Each encounter was evaluated using the scoring method described in section 3.4. All but two occurred or were simulated to represent the airport environment. Therefore, the terminal area Hazard Zone and Non-Hazard Zone definitions were used during the evaluation of those encounters. For the other two encounters, which occurred en-route, below 10,000 ft MSL criteria was used. If there was no alert issued, the encounter was analyzed as to whether an alert would have been necessary. If one or more alerts were issued, all alerts were analyzed but only the first was used to calculate alert time before CPA.

7.2.2 Overall Prototype Alerting Statistics

There were a total of 365 encounters throughout the flight test program. For those 365 encounters, the prototype issued 532 alerts, which is an average of 1.46 alerts per encounter. Table 7-1 shows a summary of the aggregate performance of the TSAA prototype during the flight test program in terms of the same performance metrics used for the algorithm tuning in the previous chapter.

Table 7-1: Overall Prototype Performance

Performance Measure	Prototype Performance 2/22/13-5/14/13
Missed Alerts	0%
Late Alerts	0%
Nuisance Alerts (Targets of Opportunity)	35.49% (34 of 92 alerts)
Average Alert Time before CPA	31.3 sec (SD: 18.3 sec)

It should be noted that throughout the flight test, the prototype suffered system malfunctions and in some occasions even had to be re-booted in flight. If an encounter

occurred during such a system malfunction, it was not considered in this analysis. Also, an alert that was missed during an encounter as a result of a system malfunction was not counted against the system.

7.2.3 NUISANCE ALERT PERFORMANCE DURING FLIGHT TEST ("WRAP AROUND" ALERTS)

Over the duration of the flight test program, 92 unplanned encounters with targets of opportunity occurred. The other encounters, which were with known targets, were designed to have a CPA in the may alert zone to force alerts. As a result, only alerts issued on targets of opportunity were used to evaluate the nuisance performance of the TSAA prototype. Also, since the flight tests were designed to solicit as many encounters as possible, a nuisance alert *rate* calculation would not be representative of what the TSAA performance would be during real-world operations, hence the use of percentage in Table 7-1.

On the 92 targets of opportunity, the TSAA prototype issued 34 nuisance alerts (nuisance alert percentage, 35.49%). Figure 7-6 shows the CPA for each one of the 34 nuisance alerts issued on targets of opportunity as well as the size of the Non-Hazard Zone for the terminal area. If the CPA had been within the Non-Hazard boundary, the alert would not have been scored as a nuisance. The nuisance alerts are differentiated into three categories. Nuisance alerts shown with a blue diamond were encounters in which, the pilot took an evasive action was taken after an alert was issued that caused the CPA to be outside of the Non-Hazard criteria. Though these ten alerts are objectively scored as nuisance alerts, they occurred due to the evasive action and are in fact instances where the system worked as designed.

Red squares represent nuisance alerts that occurred while the own-ship was operating in the airport pattern (21 total), and the three green triangles are alerts that do not fit into the previous two categories. Nuisance alerts are generally within the Non-Hazard criteria in the vertical dimension but outside by about 1,000 ft in the horizontal dimension.

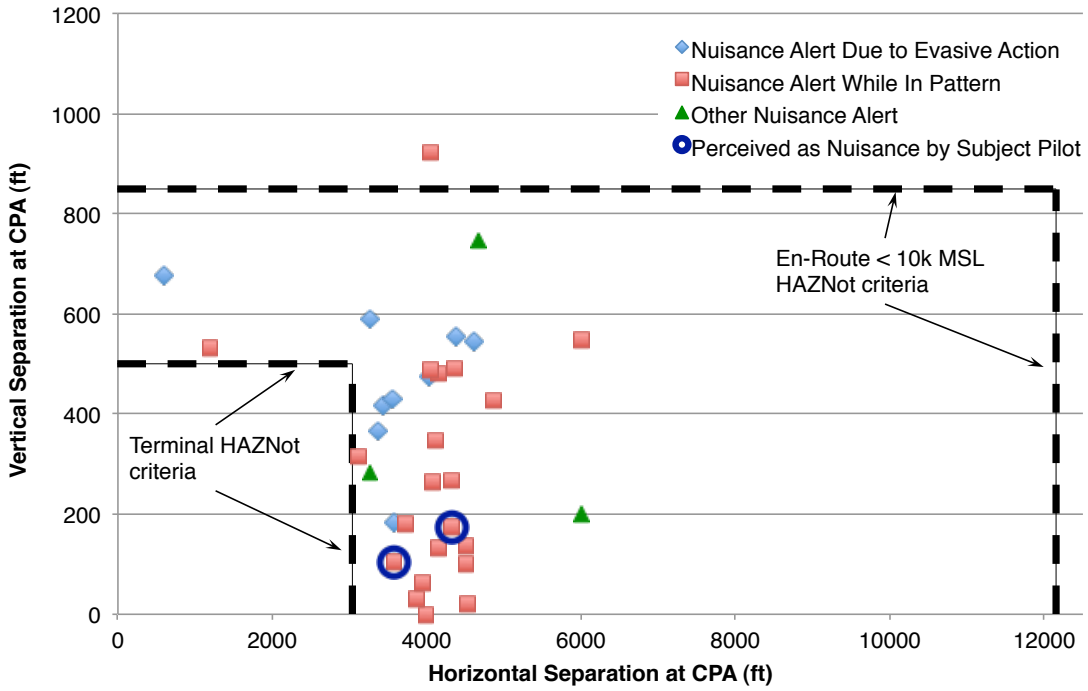


Figure 7-6: Closest Point of Approach for 34 Nuisances Issued by TSAA Prototype During Flight Test

The blue circles in Figure 7-6 are the alerts that were *subjectively* scored as nuisance alerts by the subject pilot. For all the other alerts, the subject pilot did *not* perceive the alert to be a nuisance. A conclusion that may be drawn from this is that the scoring thresholds for the Hazard Zone and the Non-Hazard Zone are set too conservatively for the terminal environment. In Figure 7-6, if the horizontal dimension of the Non-Hazard Zone were set at 5,000 ft, the total number of alerts that are scored as nuisances objectively would be reduced significantly.

An additional observation from the flight test was that the alerts scored as nuisances in Figure 7-6 frequently occurred in the airport pattern and on aircraft in close proximity to the own-ship that never really posed a threat. An example situation would be a trailing or leading aircraft operating in the same airport pattern. The alert would turn off after 7 seconds, or the minimum allowable alert duration. Out of the 21 nuisance alerts shown above, 20 fell into this category. An additional relevant observation was that this type of alert occurred most frequently when either of the two aircraft was in a steep turn.

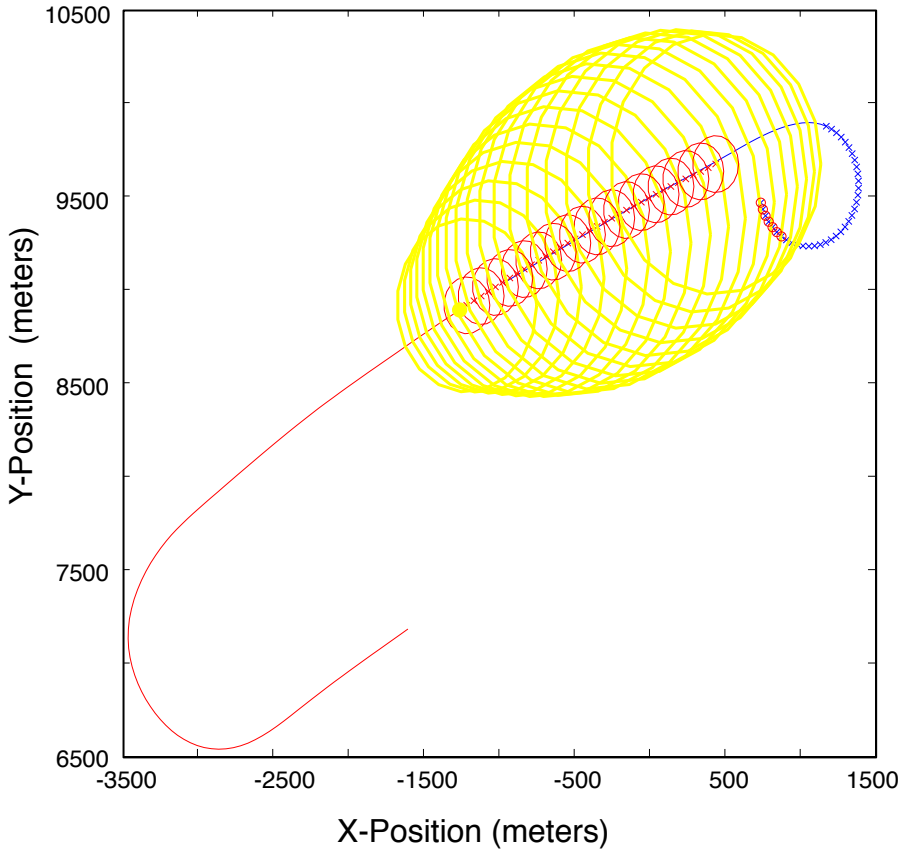


Figure 7-7: Visualization of a "Wrap-Around" Alert

Figure 7-7 shows a freeze-frame visualization of the internal algorithm workings during such an alert. The solid lines represent the historical track of the aircraft (red is target, blue is own-ship) and the x-marks represent the trajectory's discrete points in time for the respective aircraft. The size of the target's PAZ (yellow) and the size of the CAZ are drawn for every third point along the trajectory. This particular situation occurred when the own-ship was leading the target during a flight in the airport pattern. As Figure 7-7 shows, the trajectory of the own-ship is wrapped around to where it intersects the trajectory of the target behind the own-ship. Termed a "Wrap-Around" alert, this algorithm behavior is related to the trajectory wagging mentioned in 6.3.2; the next chapter will address this in more depth.

7.2.4 Average Alert Time for Alerts Issued during Flight Test

Figure 7-8 plots a histogram of the time before CPA for the first alert issued on a given encounter.

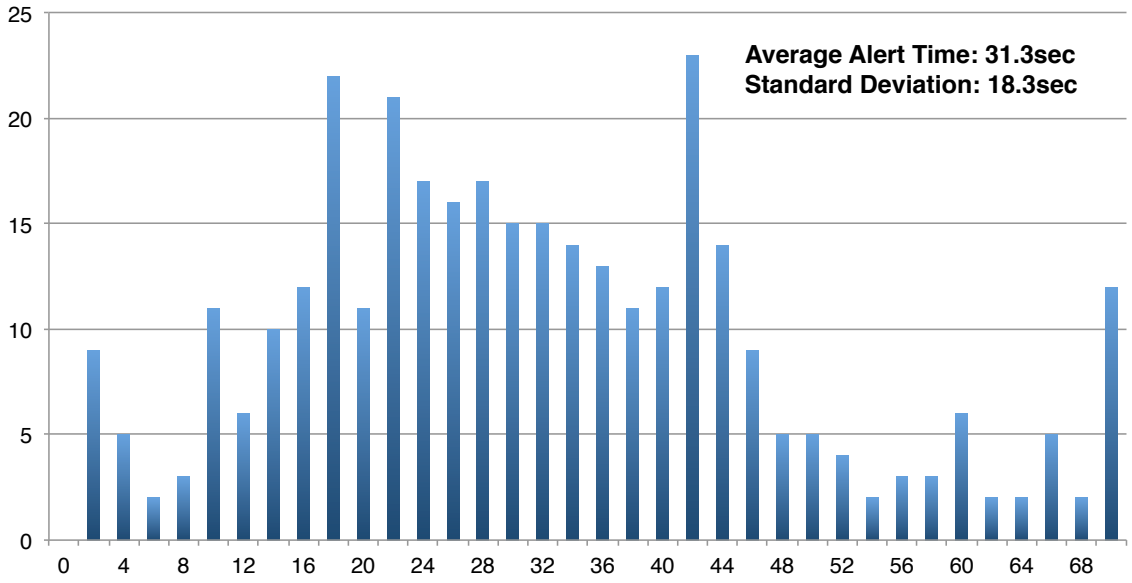


Figure 7-8: Histogram of Alert Time before CPA for all 365 Flight Test Encounters

As noted before, no missed or late alerts occurred during encounters in which the system was fully functional. However, as Figure 7-8 and Table 7-2 demonstrate, some alerts were issued with less than 12.5 seconds before CPA. Those were alerts occurred when the CPA was in the may-alert zone and thus an alert was not required. As discussed before, an alert is considered late only if the alert time is less than 12.5 *and* the CPA lies in the Hazard Zone.

Table 7-2: Summary of Missed, Late and May-Alerts with < 12.5 seconds alert time

Targets Of Opportunity	
Total Required Alerts	**1**
Missed Alerts	0
Late Alerts	0
May-Alerts with < 12.5sec	5

All Encounters	
Total Required Alerts	**43**
Missed Alerts	0
Late Alerts	0
May-Alerts with < 12.5sec	33

7.3 Comparison of Flight Test Performance to the Performance Expected from Simulation

In order to compare the performance of the prototype unit to the performance that would have been expected per the simulation, the logged flight tracks from the flight tests were

run through the simulation tool. The alerts issued during the flight test were then compared to those issued by the simulation tool, and, if significant differences were observed, the source for the difference was identified.

The objectives of this comparison were two-fold. First, the algorithm's alerting behavior was to be evaluated to ensure that the exemplar TSAA algorithm was implemented correctly. Second, the comparison was to validate the simulation environment to show that the simulated performance was in fact representative to the performance during real-world operations. More specifically, referring back to the simulation tool diagram in Figure 5-5, once it is verified that the algorithms are the same and the encounters are representative of those seen during normal operations, the only difference between the implementations will be the source of the errors present in the state data that are fed into the algorithm. Therefore, this comparison provides insight into how realistic the simulation tool's error models are.

7.3.1 NOTE ON DIFFERENCES BETWEEN IMPLEMENTATIONS

Aside from the limitation of the data logging method, an additional limitation of the comparing the simulated and actual performance is that the TSAA implementations in the simulation differs from the implementation of TSAA on the prototype unit.

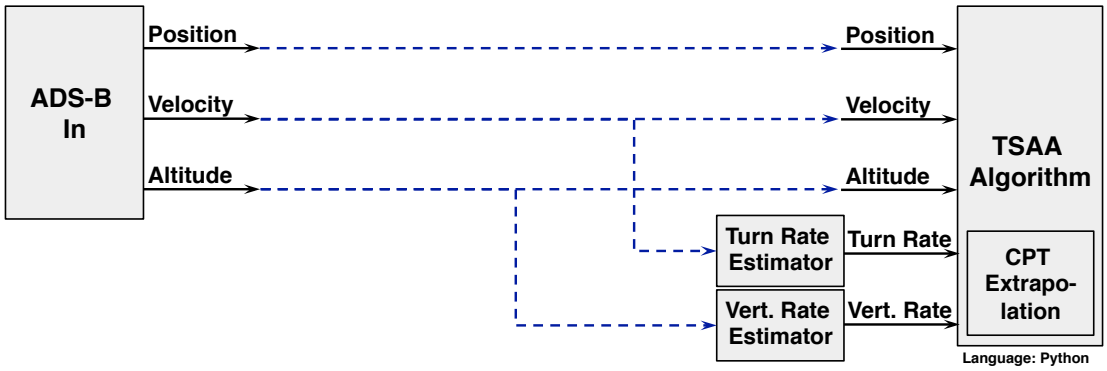


Figure 7-9: No-Tracker, Stand-Alone TSAA Implementation Used by the Simulation Tool

As mentioned in Chapter 3 and shown in Figure 7-9, the TSAA simulation tool was implemented without the DO-317A tracker to simulate the limit case of how state uncertainty could affect the algorithm performance. However, the prototype unit was implemented with a modified DO-317A tracker as shown in Figure 7-5.

7.3.2 OVERALL COMPARISON OF ALERTING PERFORMANCE

The first alerts that the two systems issued during a given encounter were compared to each other. Again, an encounter was defined as any time two aircraft were within 5,000ft or if either the prototype or the simulation issued an alert. The analysis was performed with respect to the simulation alerts; i.e., a -0.4 second difference indicates that the prototype alerts 0.4 seconds after the simulation. Table 7-3 summarizes the overall comparison.

The margin of error on the alert timing is 0.5 seconds. As mentioned, the prototype logged the data at the top of every second; as a result, independent of when the alerts were actually issued during the preceding second, they were logged at the beginning of the next second. In order to match the behavior on the prototype most accurately, the simulation was set up to call TSAA once per second but at the bottom of the second. This resulted in the simulation alerts being issued at the bottom of every second, generating a consistent offset of 0.5 seconds between the time that the prototype and the simulation alerts.

Table 7-3: Summary of Prototype To Simulation Alert Comparison

Performance Measure	Value
Average Difference in Time of First Alert	**-0.4 seconds**
With Coasting Bug (flights before 3/14/2013)	-0.9 seconds
Without Coasting Bug	0.2 seconds
Standard Deviation of Difference	**4.5 seconds**
Number of Alert Disagreements	**97 encounters**
Both Systems Alert but difference > 5.5 seconds	37
Encounters with only a Prototype Alert	26
Encounters with only a Simulation Alert	34

Table 7-3 shows that the average difference in the time at which the systems issued their alerts for all flight tests is -0.4 seconds, which is within the margin of error. Furthermore, the average differences for before and after the coasting bug mentioned in section 7.1.1 was fixed. As can be seen, the coasting bug frequently caused the prototype to alert later the simulation. Once the bug was fixed, the prototype alerted 0.2 seconds ahead of the simulation on average. Figure 7-10 is a histogram of the difference in the time of first alert between the two systems.

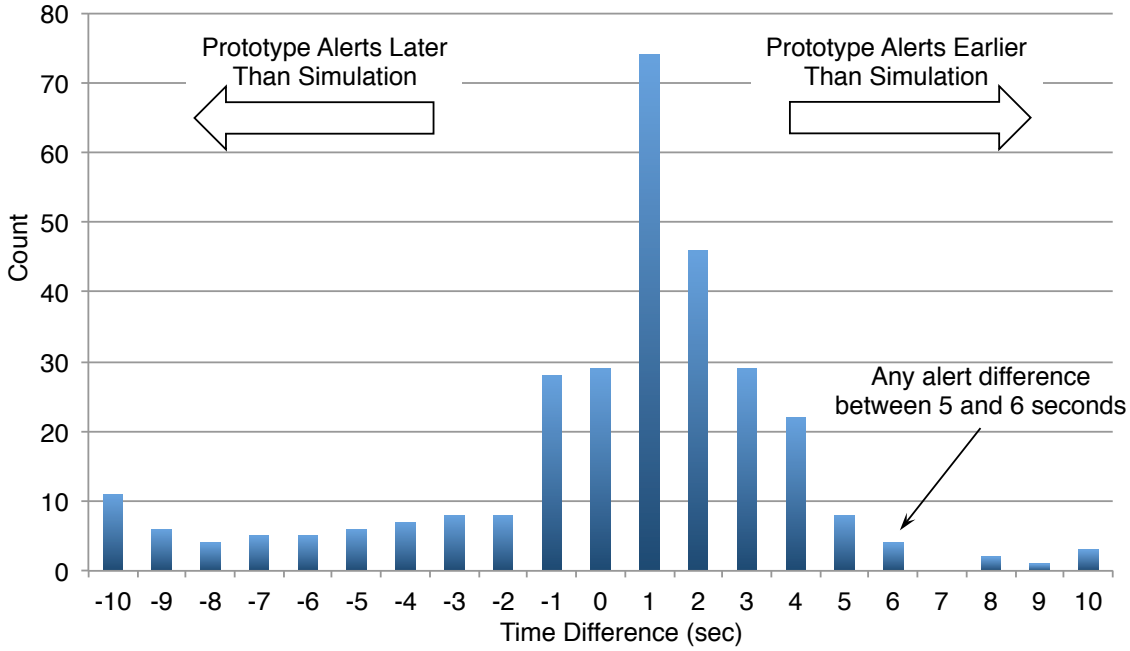


Figure 7-10: Histogram of the Difference in Time of First Alert

Table 7-3 also shows the instances that either of the two systems did not alert or that the difference in the time of the first alert was larger or equal to 5.5 seconds (one sigma in Figure 7-10). There were a total of 97 disagreements, 60 of which were examples of one system alerting. In order to better understand the source of these disagreements, these 97 alerts were analyzed in depth.

7.3.3 Analysis of Encounters with First Alert Time Difference > 5.5 sec ("Re-Acquisition Snaps")

There were 37 alert-pairs with a difference in the time of initial alert of more than 5.5 seconds. Nine of these resulted from the coasting bug, which caused the prototype to not alert until a new update became available.

Four were cases of the two aircraft operating right on the edge of the alerting threshold for a prolonged amount of time. Given the implementation differences in the timing between when TSAA actually is called combined with the slight differences in input data, the two systems calculate slightly different values in turn and vertical rate, which causes one system to alert sooner.

Twelve of the alerts were due to an effect termed the "Re-Acquisition Snap". TSAA considers a particular target as "alive" for up to 15 seconds even if no new information about it has become available for a prolonged amount of time (this is termed a data drop-out). If a target maneuvers during this time, it is possible that once new information is received and the track is re-acquired within less than 15 seconds, the track "jumps" to the most recently received data. To TSAA, this jump can look like a steep turn, which then is predicted further by the trajectory, which potentially solicits an early alert or delays an alert because the trajectory curled up due to excessive turn rates. In the prototype, the magnitude of this jump is reduced by the presence of the tracker; thus, the calculated turn rate is lower and the amount that the trajectory snaps differs. Figure 7-11 is a visualization of a severe case of a re-acquisition snap in which the high turn rate caused the trajectory to curl.

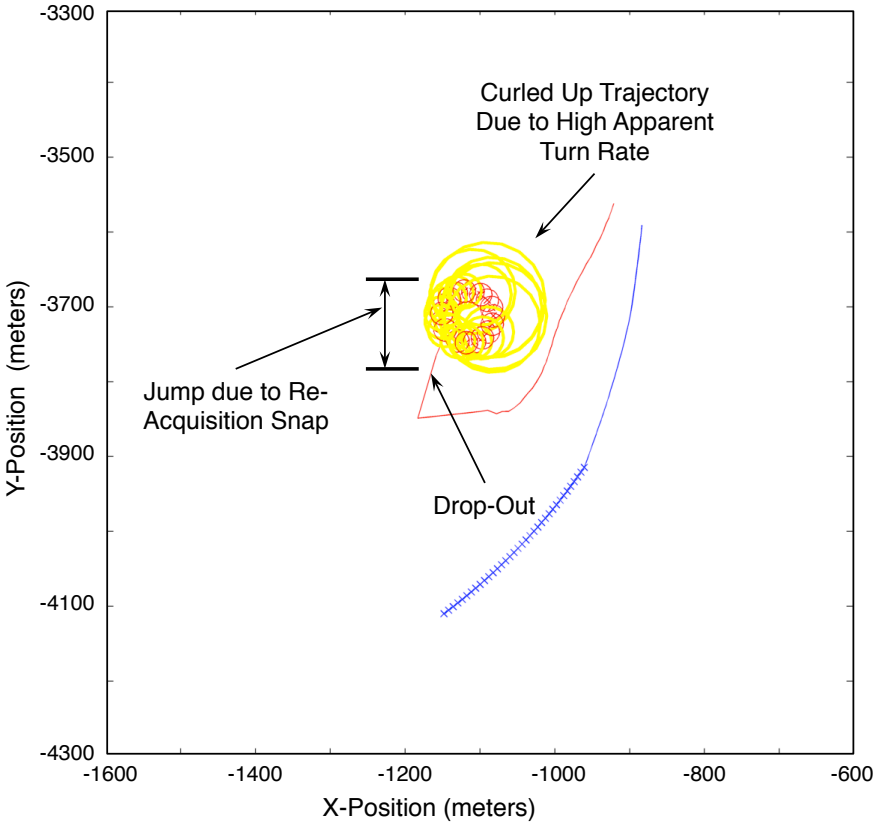


Figure 7-11: Visualization of the Re-Acquisition Snap Due to ADS-B Message Dropouts

In 12 cases, no dominant cause for the difference could be identified. However, in each of these 12 situations there was some kind of limiting circumstance that most likely influenced the alerting performance (e.g., data dropouts due to multi-path, system malfunctions, etc.).

Frequently, these situations were a combination of dropouts and two aircraft operating on the edge of the alerting threshold.

7.3.4 Analysis of Encounter where only one System Alerted

There were 60 cases in which only the prototype or only the simulation alerted. Again, each one of the alerts was analyzed individually in order to identify what was causing the difference.

As mentioned above, due to a software bug the early version of the prototype only called the TSAA algorithm when data was received from a target, which caused two instances of only the simulation alerting. Seven cases were instances of a re-acquisition snap soliciting an alert in one of the systems but not in the other. Two alerts issued by the simulation were on targets that the prototype deemed ground targets. Another 21 of the single alert cases occurred during periods of significant message dropouts caused by multi-path. 18 cases occurred during times of prolonged operations close to the alerting threshold. Just as in the previous section, the two systems calculate slightly different values in turn and vertical rate due to the differences between the implementations of when TSAA is called as well as the slight differences in input data, which causes one system to alert sooner.

In the remaining 10 cases, it is unclear what the dominant cause was for the difference. Again, in each of these, there was some kind of limiting circumstance that most likely influenced the alerting performance (e.g., data dropouts). Here also, these situations were frequently a combination of dropouts and two aircraft operating on the edge of the alerting threshold.

7.3.5 Summary Performance Comparison

In summary, the prototype performed as expected from the simulation in 73% of the encounters. In those 73%, the initial alerts were within 5.5 seconds of each other. In 24% the differences in performance could be attributed to limiting circumstantial factors such as dropouts, re-acquisition snaps, or hardware issues.

No dominant cause for the difference in performance could be identified for the remaining 3% of encounters. However, it is likely that the differences in implementation and available data in combination with circumstantial factors, such as dropouts, significantly contributed to these differences.

In light of these performance numbers it can be concluded that in general, the simulation tool generated for fast time TSAA simulations accurately approximates real-world operations. When comparing the technical performance measures, the prototype unit showed slightly better performance than what would have been expected from the

simulation in terms of nuisance alerts and late alerts (see Table 7-4). These differences most likely can be explained by the differences in implementation described at the beginning of this chapter. Considering this and the previous chapter's observations about the simulation environment, the simulation tool likely provides a conservative estimate of algorithm performance.

Table 7-4: Comparison between Flight Test Performance And Expected Airport Environment Performance

Performance Measure	Prototype Performance 2/22/13-5/14/13	Simulation Performance, Low Altitude SBS Data Set, ADS-B Nominal Target
Nuisance Alerts (Targets of Opportunity)	35.49% (34 of 92)	50.56%
Missed Alerts	0%	0.04%
Late Alerts	0%	0.04%
Average Alert Time before CPA	31.3 sec (SD: 18.3 sec)	30.6 seconds (SD: 18.4 sec)

Though the prototype performed as expected during the flight tests, certain undesirable and emergent algorithm behaviors were identified. In order to address them, a secondary analysis of the TSAA exemplar algorithm was conducted in the simulation environment (now validated); this is described in the next chapter.

Chapter 8

ADDRESSING UNDESIRABLE TSAA ALGORITHM BEHAVIOR IDENTIFIED DURING FLIGHT TEST

The following undesirable algorithm behaviors were observed during the flight test and algorithm tuning efforts:

- **Re-Acquisition Snaps (section 7.3.3):** If a target maneuvers when no data is available from it (i.e., message dropout) the reported positions and velocities that TSAA receives can be significantly different than what was received before the dropout, causing the trajectory to jump (see Figure 7-11). To TSAA, this jump can look like a steep turn, potentially causing the estimation of excessive turn rates.
- **Trajectory Wagging (section 6.3.2):** Due to small oscillation in the calculated turn rate, trajectory wagging can unnecessary solicit alerts that can last for the minimum allowable time. This also can cause a trajectory to wag in and out of alert state, causing unnecessary re-alerting during a single encounter. Trajectory wagging has two root causes: (1) noise in the state data used to calculate the turn rate and (2) operational oscillations encountered during normal operations.
- **Wrap-Around Alert (section 7.2.3):** In highly dynamic environments in which the target or the own-ship make frequent steep turns, the constant turn rate trajectory prediction can cause the trajectory to "warp around" and alert on low-threat traffic (e.g., behind own-ship). This type of alert was encountered frequently during flight tests in the airport pattern. Often the alert lasts the minimum allowable 7 seconds.

All three of these undesirable behaviors are related to the fact that TSAA uses a constant turn rate trajectory estimation and, by extension, must calculate the turn rate for the own-ship and any target of interest. In the case of the trajectory wagging and the re-acquisition snap, noise in the data used to calculate the turn rate results in an erroneous and noisy estimation of the current turn rate. In the wrap around alert, the turn rate may be calculated correctly, but when the length of the trajectory is taken into account, the algorithm predicts an unrealistic behavior for the aircraft (i.e., a full 360 degree turn over the next 30 seconds).

It is also important to note that TSAA must calculate its own vertical rate to project the change in altitude along the predicted trajectory.

To address these concerns, a natural first step is to ensure that there is minimum noise in the input data to TSAA. This is achieved by adjusting the TSAA avionics architecture, as discussed 8.1. As a second step, a more rigorous analysis of the situations and mechanisms that introduce the undesired algorithm behaviors is conducted in section 8.2. Based on the results from the analysis, an improved TSAA alerting logic is derived. The performance of the TSAA exemplar algorithm with this improved alerting logic is re-evaluated in section 8.3. Figure 8-1 is a schematic representation how the undesirable alerting behaviors will be addressed.

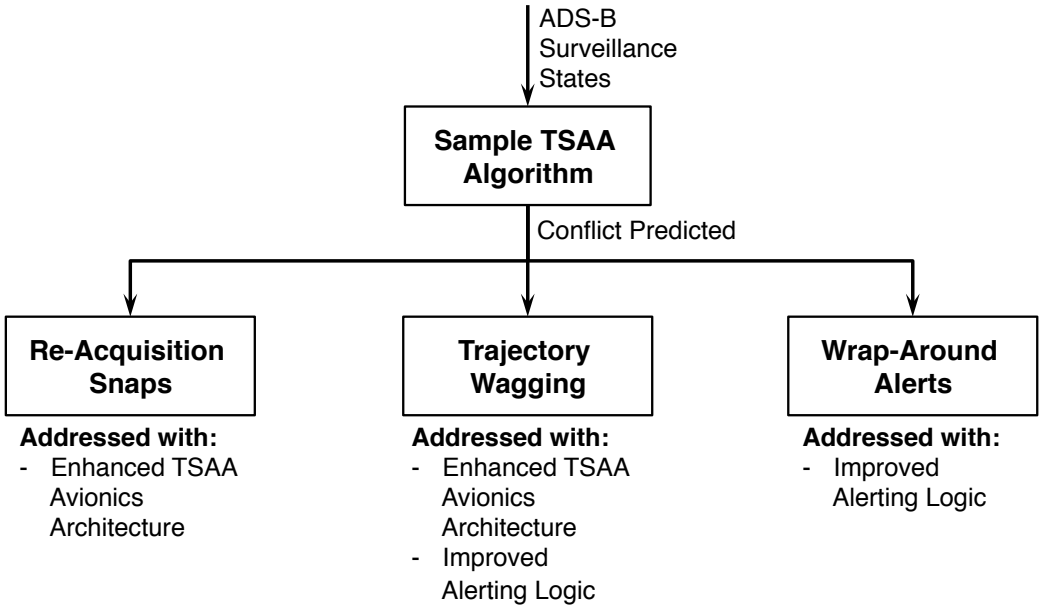


Figure 8-1: Approaches Used to Address Undesirable Behaviors of TSAA Sample Algorithm

8.1 Enhanced TSAA Avionics Architecture To Prevent Trajectory Wagging and Re-Acquisition Snaps

The exemplar algorithm as described in Chapter 3 estimates its own turn rate and vertical rate. This is necessary because the DO-317A tracker does not estimate turn rate. However, as mentioned in Chapter 3, it would be ideal if the heading and turn rate were estimated in the DO-317A tracker for a variety of reasons, two of which are addressed here. First, any other application on board the own-ship that may need to use turn rate as an input state

would have access to it. Second, the tracker already maintains the covariance matrices used to estimate position and velocity, potentially allowing for a more sophisticated approach to estimating the turn rate than what is currently implemented in the TSAA sample tracker. Additionally, were the turn rate estimate in the tracker, the extrapolation to a common point in time could be performed using that constant turn rate instead of a constant heading extrapolation.

The same logic applies for estimating the vertical rate. Currently, the tracker does estimate the vertical rate, but as an input, it requires ADS-B–reported vertical rate. However, the ADS-B message does not require the vertical rate component, and thus it is not available for all aircraft. Since TSAA needs a vertical rate estimate even if ADS-B does not report it, the additional functionality of estimating the vertical rate solely based on the reported altitudes (a required ADS-B message element) would need to be added to the tracker. Figure 8-2 is a schematic representation of this implementation of TSAA with the DO-317A tracker with enhanced vertical and turn rate estimators.

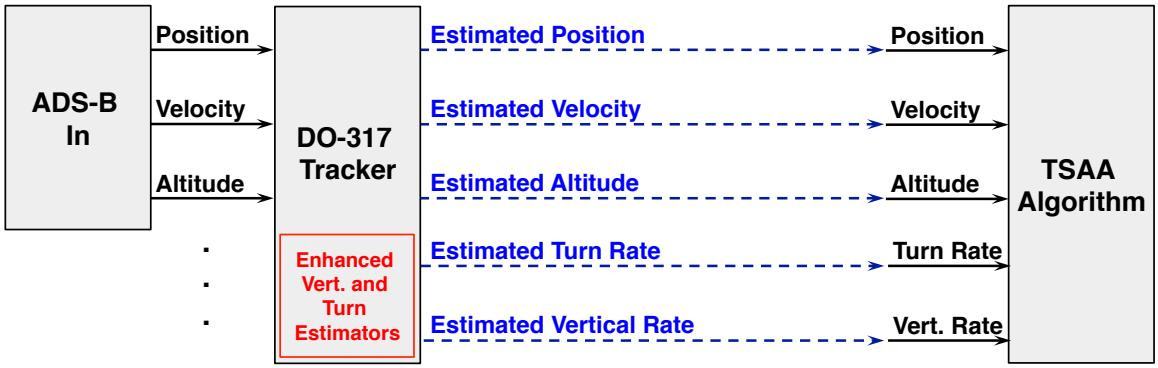


Figure 8-2: Proposed TSAA Architecture with Modified DO-317A Tracker

With this implementation, the TSAA algorithm would not perform any state estimation but rather would focus solely on determining whether a given target poses a threat to the own-ship. This approach would be in keeping with the original intention of the tracker to be a sole source of state information for all application on-board the own-ship. Additionally, this proposed system architecture would satisfy the architectural system requirement AR1.

The discussed enhancements to the DO-317A tracker have been proposed to the DO-317A standards team and are currently under evaluation for inclusion in DO-317B. It is expected that this improved avionics architecture would result in reduced noise in the input data provided to TSAA, which in turn would reduce re-acquisition snaps and noise induced trajectory wagging.

8.2 Improved Alerting Logic To Prevent Wrap-Around Alerts And Alerts Due To Trajectory Wagging

In order to better understand the mechanisms that caused wrap-around alerts and alerts/re-alerts due to trajectory wagging, alerts that were scored (objectively) as nuisances in Section 6.3.2 were analyzed again. Specifically, the state-space at the time of alert issuance was searched to identify any potential patterns that would uniquely identify the unfolding of either of the two undesirable events. Figure 8-3 shows a sample analysis for the wrap-around case. The tau values are calculated as the current horizontal and vertical separations divided by the horizontal and vertical closure rates, respectively. As such, they represent the times to the closest point if the current separation continued to decrease at the current rate.

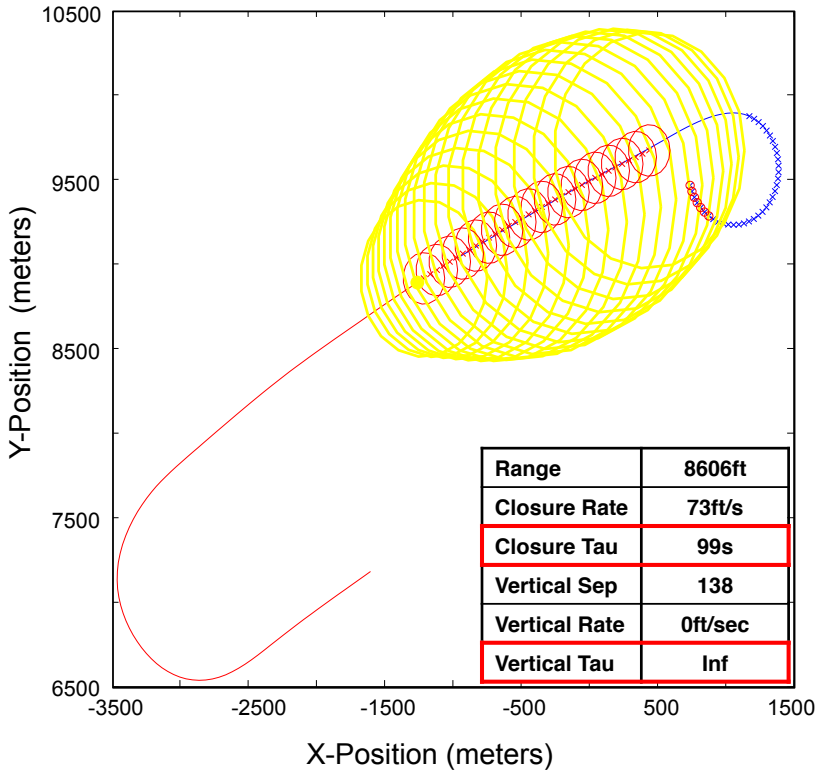


Figure 8-3: Sample Analysis of State Space Variables for A Wrap-Around Nuisance Alert

The pattern in Figure 8-3 is common for wrap-around alerts. Two aircraft are going in generally the same direction in the airport pattern. As a result, the geometry of the two aircraft at the current time is low-threat. During the process of flying the pattern, however, both aircraft will eventually perform somewhat sharp, 90-degree turns. During this turn the

constant turn rate trajectory predicts an unrealistic full 360-degree turn. Though a conflict is predicted by the constant turn rate trajectory, no conflict is predicted when evaluating the state space at the current time. Rather, in the horizontal dimension, the closure rate is such that the target would require 99 seconds to catch up to the own-ship while in the vertical dimension the two aircraft will be never actually reach co-altitude. In other words, the two aircraft are separating vertically and closing very slowly horizontally.

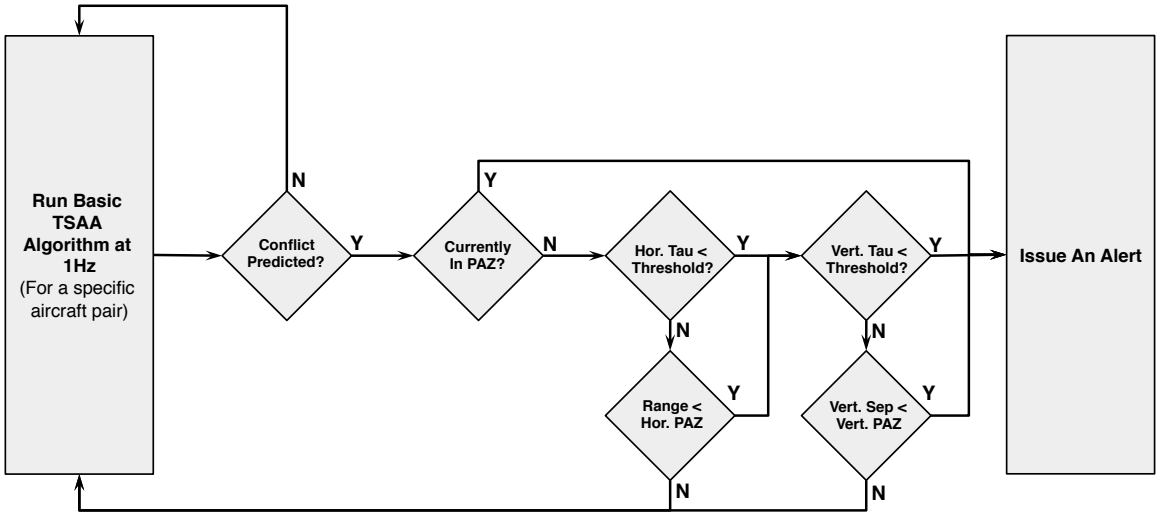
Using the approach as described above, the re-alerting caused by trajectory wagging occurs in situations in which an encounter is ongoing (i.e., the two aircraft are closing on each other) but the constant turn rate trajectory is no longer predicting a conflict. As a result, the alert is turned off only to be re-issued a few seconds later when the trajectory swings back and predicts a conflict again. In reality, the encounter never resolved but the trajectories predicted *future* behavior that would have resolved it.

8.2.1 DERIVATION OF ADDITIONAL TSAA LOGIC

Given the observation from state space, a secondary alerting logic was implemented to the basic TSAA algorithm. Described in pseudo code, the secondary logic can be described as follows:

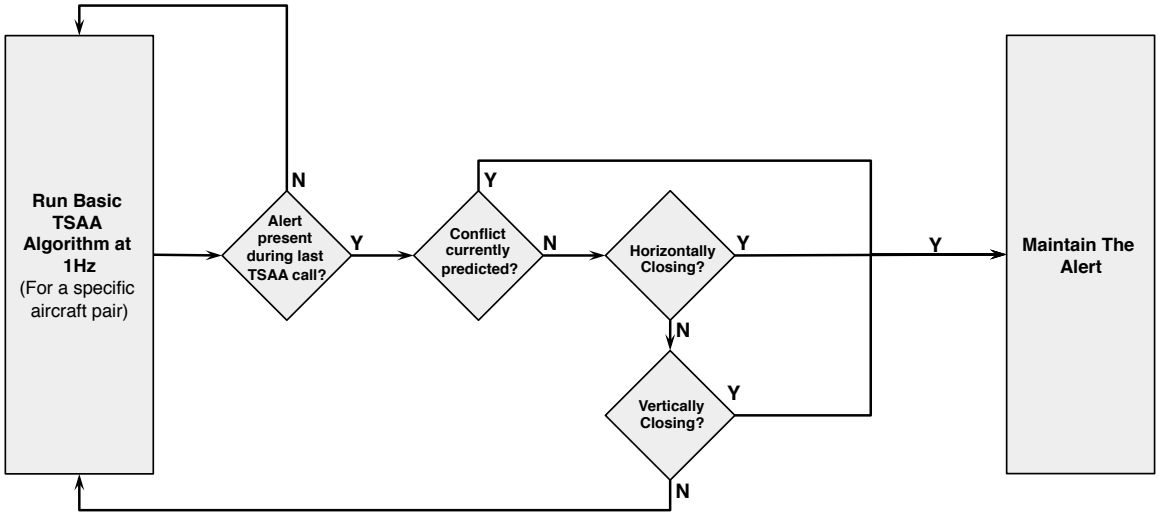
- **Wrap-Around Alert Prevention Pseudo Code:** Unless the own-ship is currently inside the PAZ, do not let an alert be issued if the currently observed states do not indicate that the closest point of approach will be reached within the time that TSAA predicts into the future (i.e., CPA is farther out than [look-ahead time + PAZ scaling] seconds)
- **Re-Alert Prevention Pseudo Code:** Do not let an alert be turned off unless the target aircraft is no longer closing on the own-ship in either the vertical or the horizontal dimension

Figure 8-4 and Figure 8-5 show the flow diagrams of the additional logic.



Note: The Tau Threshold is defined as the look-ahead time plus the PAZ scaling factor in the respective dimension.

Figure 8-4: Secondary TSAA Logic Used To Identify Wrap-Around Alerts



Note: "Closing" is defined as a positive closure rate horizontally and a positive relative velocity vertically

Figure 8-5: Secondary TSAA Logic Used to Prevent Re-Alerts Due To Trajectory Wagging

8.3 Re-Evaluating TSAA Sample Algorithm Parameter Combination with New Logic

Introducing new logic to TSAA may cause the previously tuned algorithm parameters to no longer be the best possible combination to obtain the desired performance characteristics. Therefore, the performance for the 100 hypercube points evaluated in Chapter 6 were re-evaluated. Figure 8-6 and Figure 8-7 show the differences in the performance of the 100 Hypercube Points first without and then with the new logic for the nominal ADS-B target.

As the pattern of the dots' shift almost exclusively to the left shows, adding the improved logic removed a significant number of nuisance alerts. Also, in general the sizes of the dots are reduced in size, which indicates that the total number of alerts is reduced.

When the parameter trade-off analysis described in Chapter 6 is repeated for each parameter, the performance variability percentages that can be explained by a given parameter (sensitivity) changes slightly but the curve shapes remain the same. By extension, the optimal selection of a given parameter remains the same as well. As a result, the tuned parameters were not changed after the new logic was introduced.

The alert duration histograms in Figure 8-8 through Figure 8-10 demonstrate the improvement in alert duration – the metric that initially indicated the presence of frequent short-duration alerts in Chapter 6.

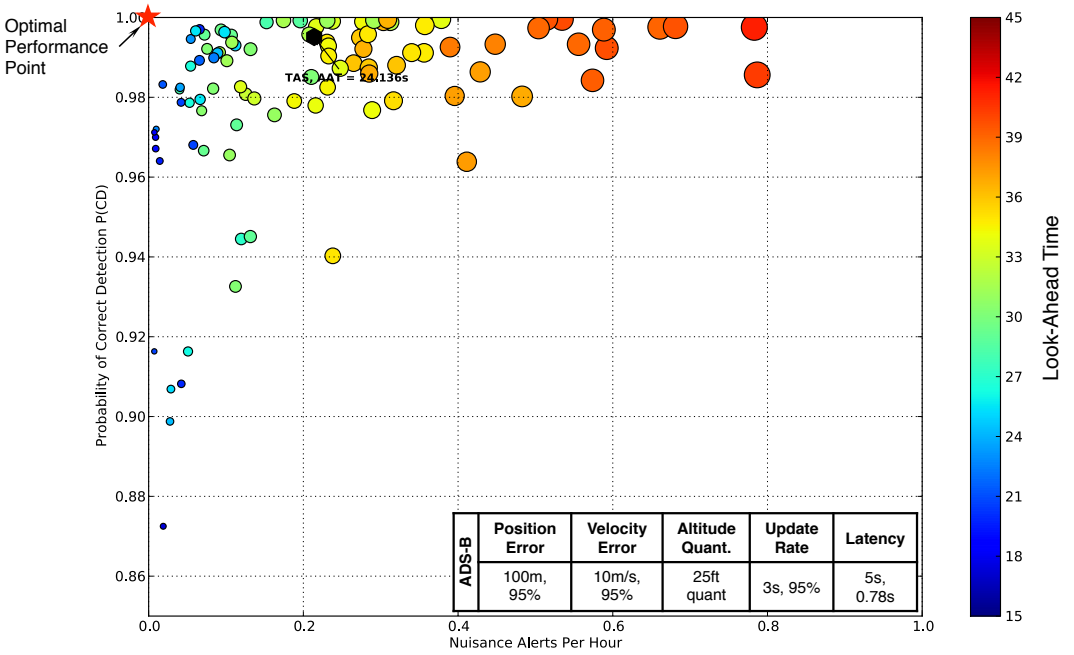


Figure 8-6: 100 Parameter Hypercube Points for TSAA WITHOUT the Additional Logic

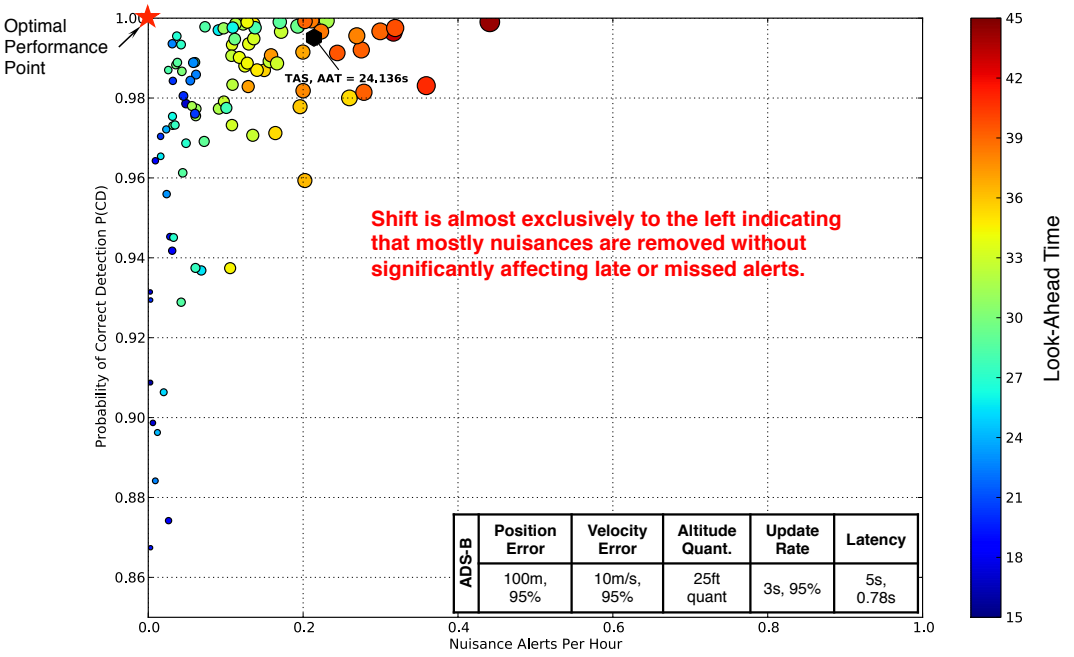


Figure 8-7: 100 Parameter Hypercube Points for TSAA WITH Improved Logic

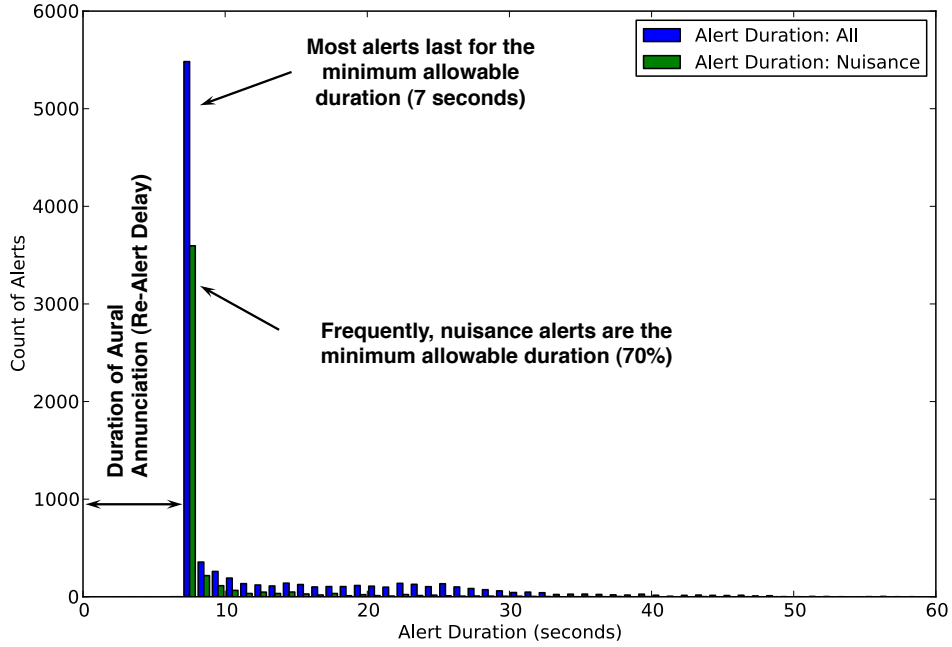


Figure 8-8: Histogram of Alert Duration of Alert Issued WITHOUT New Logic (Nominal ADS-B Target)

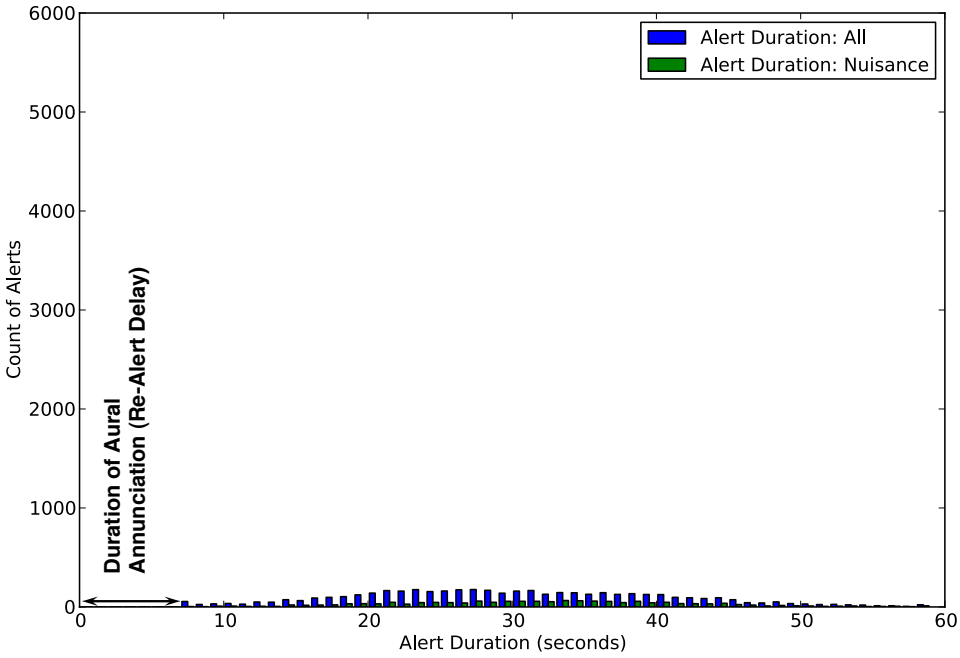


Figure 8-9: Histogram of Alert Duration of Alert Issued WITH New Logic (Nominal ADS-B Target)

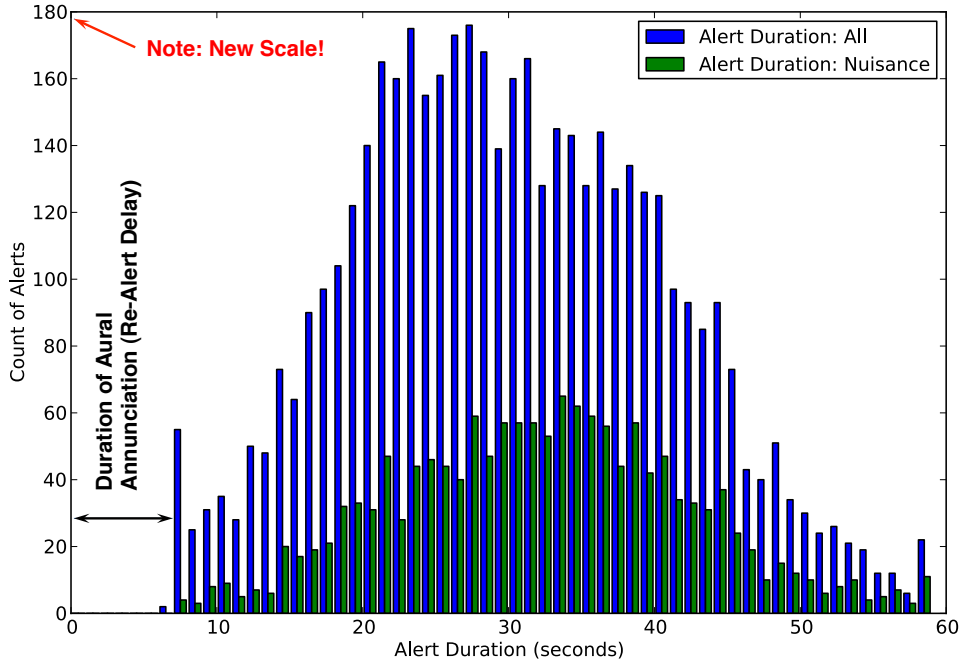


Figure 8-10: Zoomed Histogram of Alert Duration of Alert Issued WITH New Logic (Nominal ADS-B Target)

Table 8-1 and Figure 8-11 show the values comparing the performance with and without the new logic in terms of performance metrics (the same normalization values are used as listed in Table 5-8).

Table 8-1: Performance Comparison Between TSAA With and Without the Improved Logic

Technical Performance Measure	Original TSAA Algorithm	TSAA Algorithm with New Logic
Nuisance Alert Rate (hours between nuisance alerts)	9.3	26.6
Percent Correct Detection	99.4	98.9
Percent Late Alerts	0.52	0.19
Percent Missed Alerts	0.11	0.93
Average Alert Time (seconds before CPA)	31.4	28.6
Total Number of Alerts	9646	4521

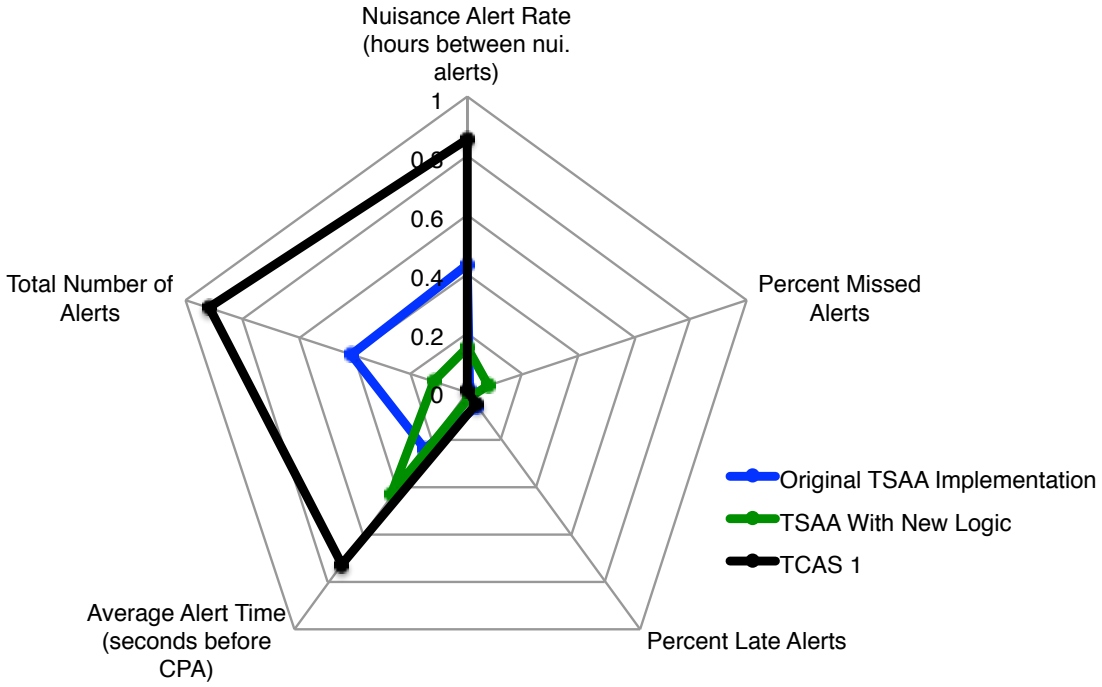


Figure 8-11: Polar Plot Visualization of TSAA Before and After Addition of New Logic (smaller is better)

As Figure 8-11 shows, TSAA with the new logic outperforms the original TSAA logic significantly in terms of nuisance alerts and total alert count. However, a small cost is associated with those gains in terms of average alert time and missed alert rate. A quick investigation into the missed alerts revealed that the additional encounters fell into the category of encounters described in section 6.4. Those particular encounters are initialized in a non-physical manner that can result in higher missed alert rates. Were some of those limitations removed, the missed alert percentage most likely would improve; nonetheless, the performance requirement of a missed alert percentage of less than 5% is still met.

Chapter 9

SUMMARY AND CONCLUSIONS

9.1 Summary of the Development of TSAA

Motivated by the introduction of ADS-B to the NAS, TSAA was developed to generate incentives for GA to equip with ADS-B avionics both ahead of the mandate and in non-rule airspace. Figure 9-1 shows the process followed.

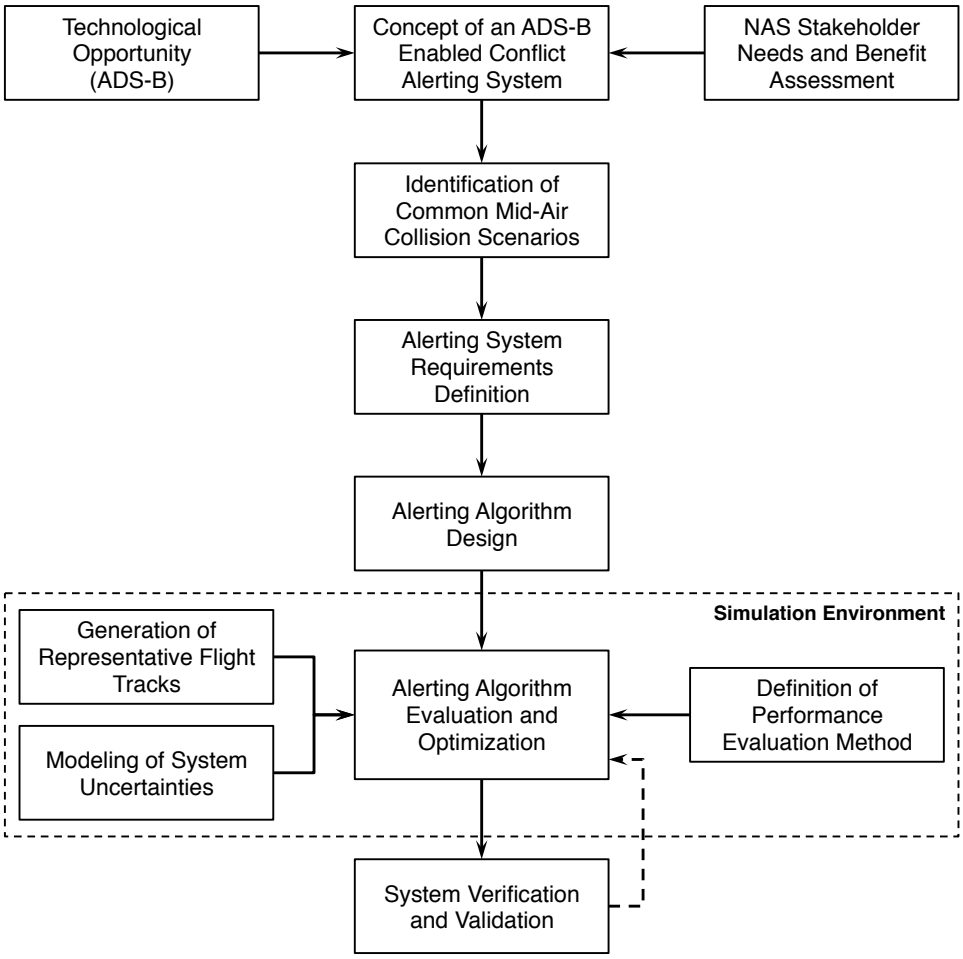


Figure 9-1: Steps Followed During the Design of TSAA

As a first step, the last ten years of mid-air collisions and near mid-air collisions were analyzed to identify areas of high benefit for conflict alerting. Combined with customer expectations and a literature review, this analysis led to the definition of the system requirements for TSAA shown in Table 9-1 through Table 9-4.

Table 9-1: Summary of Stakeholder Requirements

Requirement Number	Requirement
SH1	The equipage costs shall be kept to a minimum
SH2	TSAA shall be an alerting system and not an avoidance system
SH3	TSAA shall be able to alert during highly dynamic operations
SH4	TSAA shall be able to alert on all ADS-B targets, including TIS-B (radar derived) targets
SH5	The exemplar TSAA system be able to serve as the basis for the certification standard
SH6	The time to the introduction of TSAA shall be kept to a minimum
SH7	TSAA shall be primarily designed to operate with ADS-B targets

Table 9-2: Summary of Functional Requirements

Requirement Number	Requirement
FR1	TSAA shall reliably alert in all 14 scenario categories

Table 9-3: Summary of Architectural Requirements

Requirement Number	Requirement
AR1	TSAA shall be capable of operating within the context of a DO-317A avionics architecture
AR2	TSAA system performance shall be evaluated using an implementation without a DO-317A tracker as a conservative approach
AR3	TSAA shall be capable of maintaining alerting performance across a range of levels in target state uncertainty
AR4	TSAA shall have to equipment classes, similar to the TAS equipment classes
AR5	TSAA shall be subject to the operational, display and performance requirements defined for ATSA-AIRB

AR6	TSAA shall follow the currently existing guidance on symbol coloring for caution and warning level alerts, conform to operational procedures and regulations and be compatible with currently operational systems

Table 9-4: Summary of Performance Requirements

Requirement Number	Requirement
PF1	The performance of TSAA shall exceed the performance of a TCAS I or TAS-like system
PF2	The missed alert percentage of TSAA shall not be larger than 5%
PF3	The nuisance alert percentage of TSAA shall not be larger than 5%
PF3	The nuisance alert percentage of TSAA shall not be larger than 5%

A new algorithm was developed for TSAA based on these requirements. The algorithm's defining features are that it uses a constant turn rate trajectory prediction and adjusts the alerting threshold based on the predicted geometry and closure rate along the trajectory.

Internal to the algorithm, 14 parameters control how the algorithm predicts the trajectories, how alerting thresholds change, and ultimately when the algorithm issues alerts. Depending on how these parameters are set, the algorithm will perform differently given a particular level of state uncertainty or particular operational environment. In order to obtain the desired performance outlined by the system requirements, a method was developed to tune the performance of the algorithm.

Next, a comprehensive flight test program was conducted using the tuned algorithm. The program consisted of three phases: engineering check-out of the prototype avionics, human factors evaluations, and high performance and helicopter testing at the FAA Tech Center. The algorithm alerted as expected from the simulations and was positively received by the subject pilots during the human factors evaluations.

During the flight test and the algorithm tuning, three undesirable algorithm behaviors were identified: re-acquisition snaps, trajectory wagging, and wrap-around alerts. Introducing an improved avionics architecture and secondary alerting logic improved TSAA's nuisance alerting performance by a factor of three without significantly affecting other performance metrics.

9.2 Major Components of the Development Effort

A set of self-contained components and methods were introduced as part of the development process for TSAA.

9.2.1 TSAA Exemplar Algorithm As A Future Certification Standard

An exemplar algorithm was developed to serve as the basis for the development of the international certification standard. Once the standard is finalized and published, the algorithm also serves as a means of complying with the standard. The algorithm is distinct in its simplicity: by predicting discrete trajectories and alerting based on buffer zones, it maximizes the precedent set by the TCAS algorithm, the current industry standard. However, the TSAA exemplar algorithm expands on this precedent by including turn rate and calculating the predicted closure rate to size the airspace buffer zone. Figure 9-2 is a conceptual diagram of how the exemplar TSAA algorithm propagates trajectories and calculates airspace buffer zones based on the predicted geometry. In combination with the higher quality data available via ADS-B, this algorithm allows for a significant improvement in nuisance alert rates and average times of alert before CPA. Additionally, the exemplar algorithm allows for reliable alerting during operations in high-density environments, such as the airport, where current systems are of limited usability.

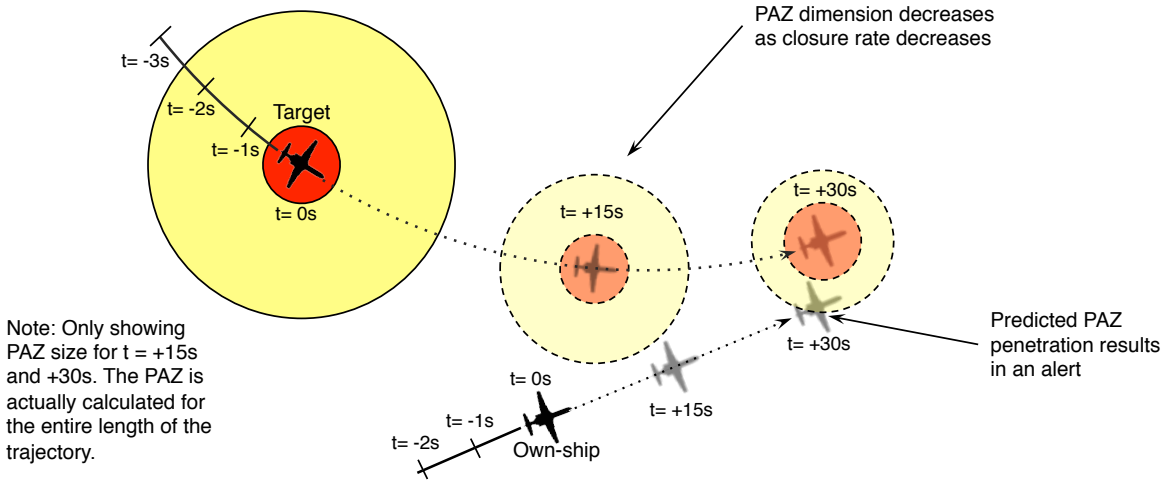


Figure 9-2: Schematic Representation of Alerting Logic Combining Protected Airspace Zones and Constant Turn Rate Trajectory Prediction

9.2.2 Development of a Method To Tune Alerting Systems

The exemplar TSAA algorithm contains 14 internal parameters that define its alerting behavior. These 14 parameters needed to be tuned to obtain the desired algorithm

performance. During the development and tuning of the TSAA algorithm, the algorithm's performance was measured using five competing metrics: the nuisance alert rate, the missed and late alert rates, the average time of alert before CPA, and the total number of alerts. Stated more generically, in order to tune the TSAA algorithm's performance, the 14-dimensional algorithm parameter space was mapped to the 5-dimensional alerting performance space. This allowed the trade-offs and relationships between the various algorithm parameters and performance metrics to be identified and evaluated.

To do this efficiently, a method was developed that enabled this type of evaluation and tuning of a complex system with multiple internal parameters and multiple competing performance attributes. The method consists of the three high-level steps shown in Figure 9-3. The first step is to use the Latin hypercube sampling approach to efficiently sample the *n*-dimensional parameter space for a set of parameter combinations spanning the entire space. The second step is to simulate a representative set of airborne encounters to evaluate the algorithm's alerting behavior given the various parameter combinations. In order to do so, a set of airborne encounters, a suite of error models, and a performance evaluation method were introduced; as discussed in sections 9.2.3 through 9.2.5. The third step is to evaluate the performance of each parameter combination in the *m*-dimensional performance space. For this step, the ROC performance visualization method proposed by Kuchar was extended to show the 5 dimensions of the TSAA performance space and allow for the visualization of a large number of parameter combinations. Using the data in the performance space, a surrogate model using the High Dimensional Model Representation (HDMR) approach was fit, and the parameters tuned to obtain the desired performance.

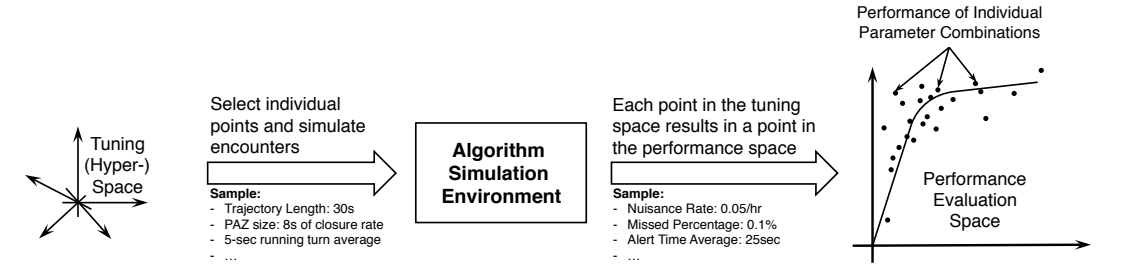


Figure 9-3: Algorithm Tuning Method

9.2.3 COMPREHENSIVE CHARACTERIZATION OF ADS-B SURVEILLANCE UNCERTAINTY

Once commissioned, TSAA will encounter varying levels of state uncertainty associated with the data it receives. Therefore, the development of TSAA required evaluating the effect of state uncertainty on the performance. To accurately model this uncertainty's effect on the performance of TSAA during development, an in-depth analysis of the sources of uncertainty in the overall ADS-B surveillance system was conducted. Based on the results, a suite of five independent error models was created to inject representative levels of uncertainty during the simulation of airborne encounters during the second step in Figure 9-3.

One of the more complex sources of uncertainty is the uncertainty introduced by the compensation of data latency. It is introduced via three different mechanisms, as shown in Figure 9-4. The first mechanism is that when latency is compensated by airborne as well as ground systems, a constant heading extrapolation is used. In highly dynamic environments such as the airport, this type of extrapolation can result in cross-tracker errors. The second source of error results from the fact that the velocity used for the extrapolation itself contains state uncertainty, which can result in over- or under-compensation along the fight track. Lastly, since the total amount of latency in the system is not always exactly known, additional over- or under-compensation may occur.

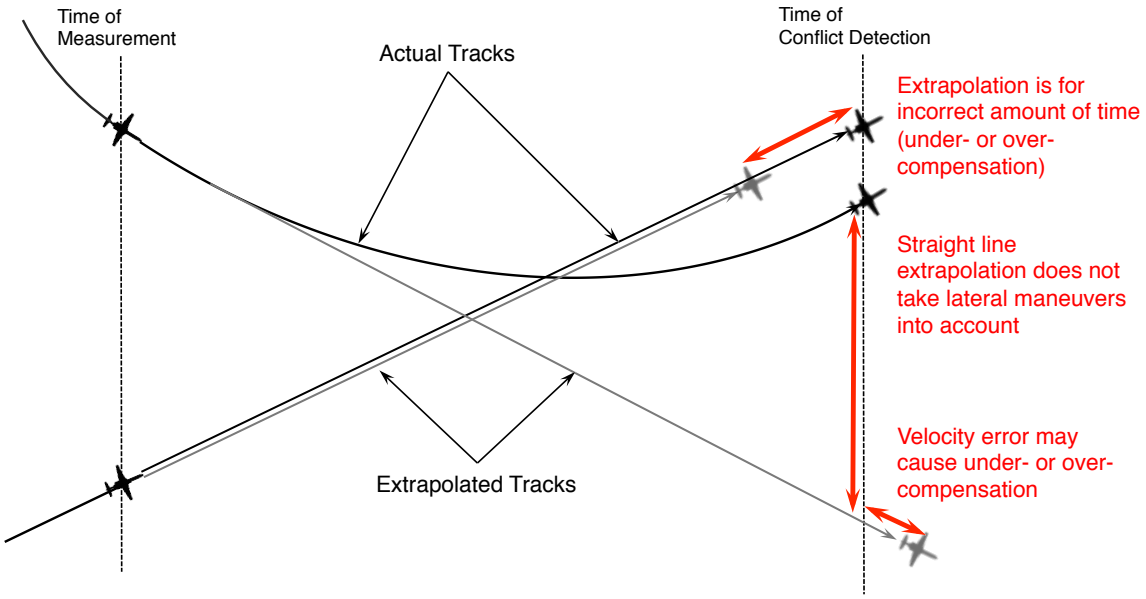


Figure 9-4: Schematic Representation of Error Sources Introduced by Latency Compensation

Also implemented in addition to the latency error model described here were error models for position, velocity, and altimetry. Lastly, a probabilistic model of the likelihood that an ADS-B message is received by the own-ship was also developed in order to model the effects of dropped messages.

Together, these models represent a suite of tools that can generate errors specific to any type of ADS-B target (i.e., air-to-air ADS-B, ADS-R or TIS-B) and accurately represent the errors that an ADS-B–enabled conflict alerting system would encounter during real-world operations.

9.2.4 Approach to Score Alerting System Performance

To score the TSAA algorithm performance per the second step in Figure 9-3, a new method for alert scoring was introduced. The method defines three own-ship centric zones according to the level of hazard that would be present if an aircraft were to penetrate that zone. Figure 9-5 shows the three zones: The red "Hazard Zone" represents the zone in which target penetration most certainly would represent a hazard. An alert would be required in such an encounter. On the other hand, aircraft that remain in the zone shown in green are considered to never pose a threat to the own-ship and thus an alert should not be issued. If the system does issue an alert in this situation, that alert is scored as a nuisance alert. In the white zone between the Hazard Zone and the Non-Hazard Zone, an alert is not required but if one is issued it is not counted against the system.

In order to determine the sizes of the various zones, this scoring approach was presented to a focus group of pilots. Table 9-5 shows the results of this survey for three different environments. TSAA performance was evaluated using this approach.

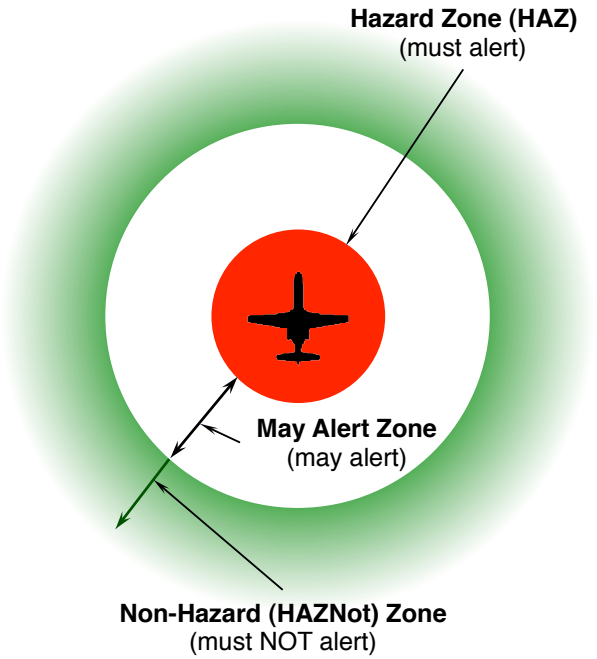


Figure 9-5: Zones used in Alert Evaluation

Table 9-5: Size of Zones used for Alert Evaluation

	Hazard Zone		Non-Hazard Zone	
Environment	**Vertical**	**Horizontal**	**Vertical**	**Horizontal**
Terminal	200 ft	500 ft	500 ft	1/2NM
En-Route (< 10,000MSL)	450 ft	500 ft	850 ft	2NM
En-Route (> 10,000MSL)	450 ft	500 ft	850 ft	3NM

9.2.5 EXPANSION OF KUCHAR'S VISUALIZATION OF THE PERFORMANCE SPACE

Various algorithm parameter combinations were simulated and then evaluated in terms of five technical performance metrics. The five metrics define a five dimensional performance space. Previous work by Kuchar proposes a performance trade-off method based on the ROC curve for evaluating various thresholds on alerting systems. This ROC curve method was adapted for the visualization of the performance of a large set of different algorithm parameter combinations in all five performance dimensions. Figure 9-6 shows the performance of the TSAA algorithm for a rule-compliant own-ship and ADS-B target for 100 different algorithm parameter combinations.

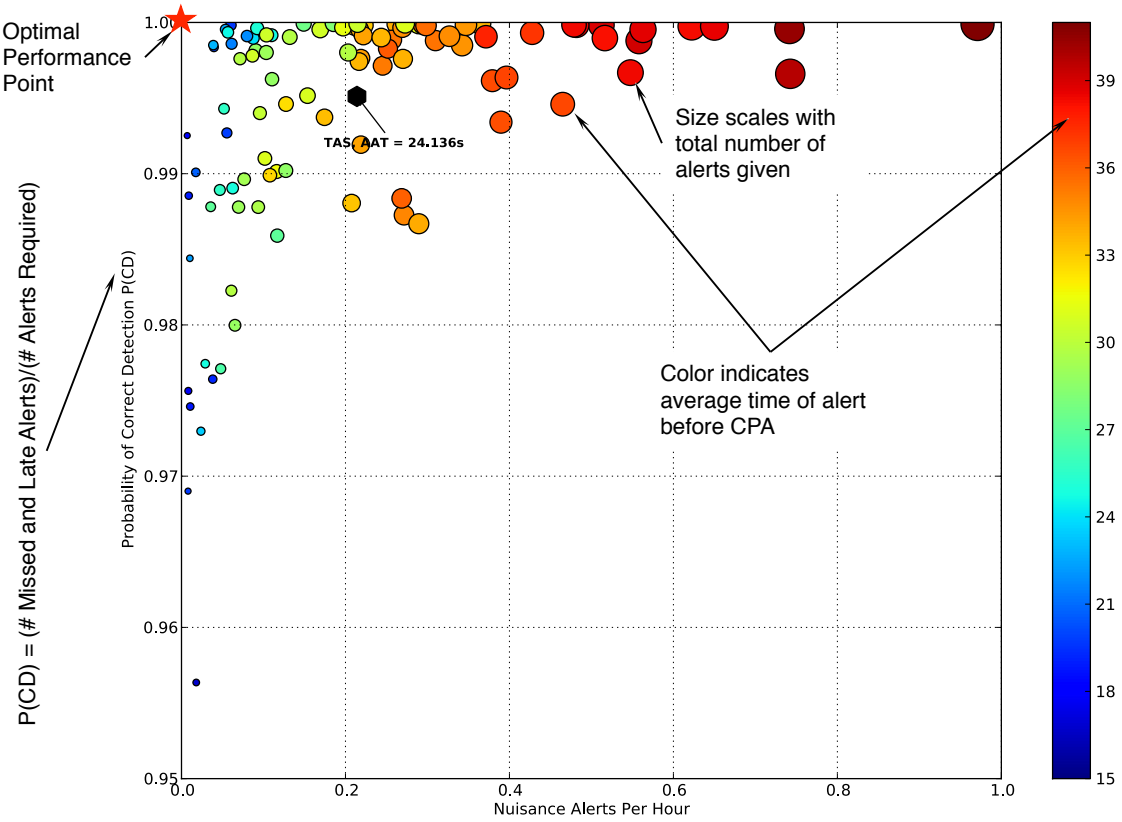


Figure 9-6: Extension of Kuchar's ROC Curve Approach for the Visualization of all Five Performance Metrics Used During the Development of TSAA

9.3 Conclusion

A new conflict alerting algorithm for TSAA has been developed, optimized, and tested in this thesis. Compared to current systems, the nuisance alert rate is more than 5 times lower, while maintaining comparable or better performance across all other performance metrics. Enabled by this reduction in nuisance alerts, the algorithm for the first time introduces reliable alerting to the airport environment, where over the last 10 years 59% of all mid-air collisions occurred and current alerting systems are of limited use. As such, TSAA has the potential to improve operational safety specifically in the general aviation community, which is most likely to be involved in a mid-air collision while operating in the airport environment. Figure 9-7 and Table 9-6 compare the performance of the current alerting system standard (TCAS I) and the newly developed TSAA algorithm.

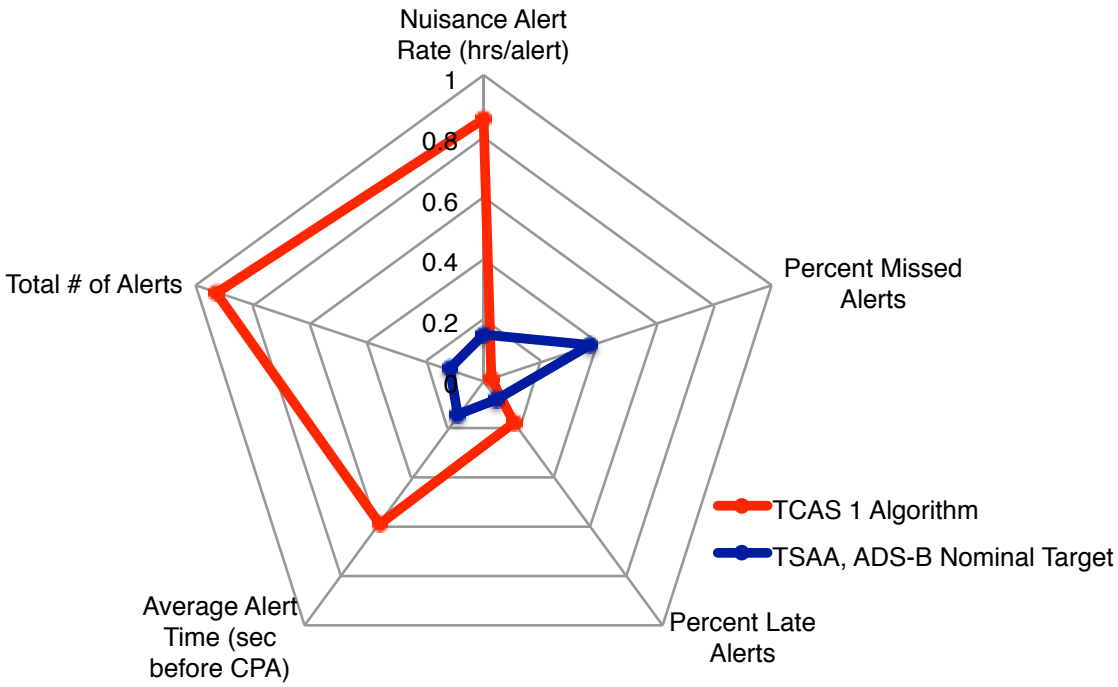


Figure 9-7: Radar Plot Comparison of Performance Between TCAS I and The TSAA Algorithm

Table 9-6: Numerical Comparison of Performance Between TCAS I and The TSAA Algorithm

Technical Performance Measure	TCAS I	TSAA Algorithm with New Logic
Nuisance Alert Rate (hours between nuisance alerts)	4.7	26.6
Percent Late Alerts	0.06	0.19
Percent Missed Alerts	0.43	0.93
Average Time of Alert Before CPA	24.1	28.6
Total Number of Alerts	18623	4521

Developing the TSAA algorithm required the characterization of the state uncertainties present in the ADS-B surveillance system. The source of error that most significantly affects the exemplar TSAA algorithm is the horizontal position error. As this thesis describes, the algorithm is very successful at protecting against horizontal position error for rule compliant ADS-B, ADS-R, and high-quality TIS-B targets. As the position information

becomes less accurate for non-compliant ADS-B targets or low quality TIS-B targets, the late and missed alert percentages begin to increase.

Given that the TSAA algorithm strongly depends on accurate turn rate estimation to propagate future aircraft trajectories, a change to the sample tracker in the DO-317A standard is proposed. Adding turn rate as an estimated state could further increase TSAA performance and introduce the possibility of conducting constant turn rate latency compensation internal to the tracker.

9.3.1 FUTURE WORK

The TSAA development effort resulted in a certifiable algorithm that will serve as the basis for the standard against which future TSAA systems will be evaluated. However, additional areas of work have been identified and are summarized here. From the perspective of the algorithm, potential future areas of work include:

EVALUATION OF BENEFIT OF DYNAMIC ALGORITHM PARAMETERS

The presented version of TSAA uses a single parameter combination for all targets. Dynamic algorithm parameter settings, however, could be used to increase the TSAA exemplar algorithm's robustness to changing levels of uncertainty. One promising approach would be to use the reported position uncertainty of the target to scale the size of the minimum protected airspace around that target dynamically.

ADAPTATION OF TSAA TO HELICOPTER OPERATIONS

TSAA as presented in this thesis was optimized for the use with fixed-wing aircraft. However, as observed by the NTSB, helicopter operations tend to have a higher risk of mid-air collision per flight hour due to frequent close proximity operations. The current implementation of the TSAA algorithm is usable for helicopter operations but may not exhibit the alerting behavior most optimal for them. One possible solution would be to evaluate the benefit of using dynamic alerting thresholds, as introduced in the previous section. Additionally, since ADS-B does provide a data field that identifies the aircraft type (i.e. fixed wing vs. rotorcraft), a type-dependent alerting logic that removes the trajectory prediction and solely alerts on proximity below a certain velocity of rotorcraft may be feasible.

IDENTIFICATION OF ON-GROUND STATUS

During the flight test, alerts were frequently issued while the own-ship was still operating on the ground. Though these alerts were not counted in the analysis, they still highlight an important issue: if TSAA is not capable of determining whether the own-ship or the target is on the ground, it may issue alerts while taxiing on the airport surface, causing a nuisance

problem. The current guidance requires any aircraft with an ADS-B installation to also be capable of determining its on-ground status. In higher-end aircraft this is frequently achieved with a "squat switch" which activates once the weight of the aircraft is on the wheels. However, on many GA aircraft, such a switch may not be installed or, if required, be a financial dis-incentive for users to install ADS-B and TSAA. A reliable means of determining on ground-status without a squat switch would therefore be beneficial to the general aviation community.

The methods developed during the design and implementation of TSAA introduce potential areas of future work:

EXTENSION OF HYPERCUBE/HDRM TUNING METHOD

For the algorithm tuning method introduced in Chapter 5, 100 sample points were used to map the algorithm parameter space. Higher numbers of sample points could provide more insight into the algorithm's performance characteristics but also would increase the computational time required to generate those points. A trade-off analysis of how the numbers of points used trades the improved statistical power for post processing with computational load may allow for a more efficient approach to this method.

APPLICATION OF TUNING METHOD TO AN "UNCERTAINTY HYPERCUBE"

An additional application of the tuning method would be to evaluate the algorithm's sensitivity to the various state uncertainty parameters (e.g., NACp, NACv). Instead of defining the tuning hypercube with the internal algorithm parameters, the tuning hypercube would be defined using those state uncertainty parameters. This would enable the maximum acceptable levels of state uncertainty to be identified efficiently.

INTRODUCTION OF CORRELATED POSITION AND VELOCITY ERRORS

One limitation of the simulation approach is that the position and velocity errors were simulated independently. In reality, however, the two errors are intimately related. Developing a correlated position-velocity error model would allow for a more accurate analysis of how the performance of the TSAA algorithm changes in the presence of state uncertainty.

BIBLIOGRAPHY

[1] F. Kunzi and R. J. Hansman, "Mid-Air Collision Risk and Areas of High-Benefit for Traffic Alerting," in *11th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference*, 2011.

[2] A. Pritchett, "Pilot Non-Conformance to Alerting System Commands During Closely Spaced Parallel Approaches," Massachusetts Institute of Technology, 1996.

[3] Federal Aviation Administration, "2010 General Aviation and Part 135 Activity and Avionics Survey." 2010.

[4] FAA, *Automatic Dependent Surveillance— Broadcast (ADS–B) Out Performance Requirements To Support Air Traffic Control (ATC) Service; Final Rule*. 2010, pp. 1–37.

[5] 2012 ADS-B In Aviation Rule Making Committee, "Report from the ADS–B In Aviation Rulemaking Committee to the Federal Aviation Administration," 2012.

[6] FAA, "ADS-B Integrated Work Plan, v2.0," 2010.

[7] 2008 ADS-B Out Aviation Rule Making Committee, "Report from the ADS-B Out Aviation Rule Making Committee to the Federal Aviation Administration," 2008.

[8] F. Kunzi, "ADS-B Benefits to General Aviation and Barriers to Implementation," Massachusetts Institute of Technology, 2011.

[9] E. A. Lester and R. J. Hansman, "Benefits and Incentives for ADS-B Equipage in the National Airspace System," Massachusetts Institute of Technology, 2007.

[10] J. K. Kuchar, "A unified methodology for the evaluation of hazard alerting systems," Massachusetts Institute of Technology, 1995.

[11] J. K. Kuchar, "Managing Uncertainty in Decision-Aiding and Alerting System Design," *Proceedings of the 6th CNS/ATM Conference*, vol. 2001, pp. 1–10, 2001.

[12] W. D. Rowe, *Understanding Uncertainty*, vol. 14, no. 5. John Wiley and Sons, 1994, pp. 743–750.

[13] H. Beyer and B. Sendhoff, “Robust Optimization – A Comprehensive Survey,” no. March 2007.

[14] S. C. Mohleji and G. Wang, “Modeling ADS-B Position and Velocity Errors for Airborne Merging and Spacing in Interval Management Application,” 2010.

[15] R. Barhydt and N. Doble, “Handling trajectory uncertainties for airborne conflict management,” *Digital Avionics Systems Conference*, 2005.

[16] 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, 2000.

[17] J. Paul, T. Venhovens, and K. Naab, “Vehicle dynamics estimation using Kalman filters,” *Vehicle System Dynamics*, no. December 2012, pp. 37–41, 1999.

[18] H. Araki, K. Yamada, Y. Hiroshima, and T. Ito, “Development of rear-end collision avoidance system,” *Proceedings of Conference on Intelligent Vehicles*, pp. 224–229.

[19] A. Vahidi and A. Eskandarian, “Research advances in intelligent collision avoidance and adaptive cruise control,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 4, no. 3, pp. 143–153, Sep. 2003.

[20] R. Alterovitz, T. Siméon, and K. Y. Goldberg, “The Stochastic Motion Roadmap: A Sampling Framework for Planning with Markov Motion Uncertainty,” in *Robotics Science and Systems*, 2007, pp. 1–8.

[21] J. Jansson and F. Gustafsson, “A framework and automotive application of collision avoidance decision making,” *Automatica*, vol. 44, no. 9, pp. 2347–2351, Sep. 2008.

[22] X. D. Cheng, Z. Y. Liu, and X. T. Zhang, *Trajectory Optimization for Ship Collision Avoidance System Using Genetic Algorithm*. 2006.

[23] I. Hwang, J. Hwang, and C. Tomlin, “Flight-Mode-Based Aircraft Conflict Detection using a Residual-Mean Interacting Multiple Model Algorithm,” *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2003.

[24] L. C. Yang and J. K. Kuchar, “Using intent information in probabilistic conflict analysis,” *AIAA Guidance, Navigation, and Control Conf*, 1998.

[25] RTCA, *DO-317A: Minimum Operational Performance Standards (MOPS) for Aircraft Surveillance Applications (ASA) System*. 2011.

[26] L. C. Yang and J. K. Kuchar, "Prototype Conflict Alerting System for Free Flight," *Journal of Guidance, Control and Dynamics*, vol. 20, no. 4, 1997.

[27] M. S. Eby and W. E. Kelly, "Free flight separation assurance using distributed algorithms," *1999 IEEE Aerospace Conference. Proceedings (Cat. No.99TH8403)*, pp. 429–441 vol.2, 1999.

[28] Y. Chiang and J. Klosowski, "Geometric Algorithms for Conflict Detection / Resolution in Air Traffic Management," *36th IEEE Conference on Decision and Control*, 1997.

[29] T. Jones, "Real-Time Probabilistic Collision Avoidance for Autonomous Vehicles, Using Order Reductive Conflict Metrics," Aeronautics and Astronautics, 2003.

[30] M. Prandini, J. Lygeros, A. Nilim, and S. Sastry, "A Probabilistic Framework for Aircraft Conflict Detection," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, pp. 1–36, 1999.

[31] M. Prandini and J. Hu, "A Probabilistic Approach to Aircraft Conflict Detection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 4, pp. 199–220, 2000.

[32] M. Prandini, O. J. Watkins, and O. Watkins, "Probabilistic Aircraft Conflict Detection," *Leonardo*, 2005.

[33] C. Munoz, R. Siminiceanu, V. A. Carreno, and G. Dowek, "KB3D Reference Manual – Version 1 . a," *System*, no. June, 2005.

[34] A. Geser, C. Muñoz, G. Dowek, and F. Kirchner, "Air traffic conflict resolution and recovery," 2002.

[35] C. Muñoz and A. J. Narkawicz, "Time of Closest Approach in Three-dimensional Airspace," 2010.

[36] H. Herencia-Zapana, J.-B. Jeannin, and C. Muñoz, "Formal verification of safety buffers for state-based conflict detection and resolution," *Proceedings of 27th International Congress of Aeronautical Sciences*, pp. 1–9, 2010.

[37] G. Dowek, C. Muñoz, and A. Geser, “Tactical Airspace Conflict Detection and Resolution in a 3-D Airspace,” *Contract*. p. 20, 2001.

[38] G. Dowek and C. Muñoz, “Conflict detection and resolution for 1, 2,..., N aircraft,” *Proceedings of the 7th AIAA Aviation, Integration, and Operations Conference (ATIO)*, no. September, pp. 18–20, 2007.

[39] V. Carreño, “Evaluation of a Pair-Wise Conflict Detection and Resolution Algorithm in a Multiple Aircraft Scenario,” 2002.

[40] M. a. Christodoulou and S. G. Kodaxakis, “Automatic Commercial Aircraft-Collision Avoidance in Free Flight: The Three-Dimensional Problem,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 7, no. 2, pp. 242–249, Jun. 2006.

[41] Eurocontrol, *EUROCONTROL Specification for Trajectory Prediction*. 2010.

[42] Eurocontrol, “Specification for Short Term Conflict Alert,” 2009.

[43] H. Hutchinson, R. Howell, and S. Turner, “Performance and safety Aspects of Short-term Conflict Alert,” 2010.

[44] H. Erzberger and R. Paielli, “Conflict detection and resolution in the presence of prediction error,” *1st USA/Europe Air Traffic Management R&D Seminar*, pp. 1–19, 1997.

[45] J. P. Chryssanthacopoulos and M. J. Kochenderfer, “Hazard Alerting Based on Probabilistic Models ⋆,” no. August, pp. 1–13, 2011.

[46] L. Yang, “Aircraft Conflict Analysis and Real-Time Conflict Probing using Probabilistic Trajectory Modeing,” Massachusetts Institute of Technology, 2000.

[47] L. Yang and J. Kuchar, “Performance metric alerting: A new design approach for complex alerting problems,” *Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 32, no. 1, pp. 123–134, 2002.

[48] FAA, *Aeronautical Information Manual*, 2012th ed. Department of Transportation, 2012, p. 736.

[49] FAA, *TSO-C147, Traffic Advisory System (TAS) Airborne Equipment*, vol. 607, no. d. Federal Aviation Administration, 1994.

[50] FAA, “Introduction to TCAS II, Version 7.1,” 2011.

[51] J. K. Kuchar and A. Drumm, “The traffic alert and collision avoidance system,” *Lincoln Laboratory Journal*, vol. 16, no. 2, pp. 277–296, 2007.

[52] D. Smith, “Traffic Alert Collision Avoidance Systems — TCAS Buyer ’ s Guide,” *Aircraft Electronics Association Pilots Guide*, pp. 26–32, 2005.

[53] FAA, *Airworthiness Approval of Traffic Alert And Collision Avoidance Systems (TCAS II), Versions 7.0 & 7.1 and Associated Mode S Transponders*, no. Tcas Ii. Federal Aviation Administration, AIR-130, 2009.

[54] Allied-Signal/Bendix Communications, “TCAS Alert and Collision Avoidance System: TCAS III,” Baltimore, MD, 1987.

[55] W. Love, “TCAS III: Bringing Operational Compatibility To Airborne Collision Avoidance,” *8th AIAA/IEEE Digital Avionics Systems Conference (DASC)*, 1988.

[56] D. Burgess and S. Altman, “TCAS III Bearing Error Evaluation.,” 1995.

[57] T. Williamson and N. a. Spencer, “Development and operation of the Traffic Alert and Collision Avoidance System (TCAS),” *Proceedings of the IEEE*, vol. 77, no. 11, pp. 1735–1744, 1989.

[58] T. B. Billingsley, M. J. Kochenderfer, and J. P. Chryssanthacopoulos, “Collision avoidance for general aviation,” *IEEE Aerospace and Electronic Systems Magazine*, vol. 27, no. 7, pp. 4–7, 2012.

[59] J. D. Griffith and W. Olson, “Coordinating General Aviation Collision Avoidance Maneuvers with TCAS Resolution Advisories.” 2010.

[60] FLARM Technology GmbH, “FLARM Installation Manual,” pp. 1–8, 2011.

[61] RTCA, *Final Report of RTCA Task Force 3 Free Flight Implementation*. RTCA, Incorporated, 1995.

[62] A. L. Mozdzanowska, R. E. Weibel, E. Lester, R. J. Hansman, and A. Weigel, “Dynamics of Air Transportation System Transition and Implications for ADS-B Equipage,” in *Technology*, 2007, no. September, pp. 18–20.

[63] SC-186 and RTCA, "DO-260B: Minimum Operational Performance Standards for 1090 MHz Extended Squitter Automatic Dependent Surveillance – Broadcast (ADS-B) and Traffic Information Services – Broadcast (TIS-B)." 2009.

[64] FAA, "Surveillance and Broadcast Services (SBS) Description Document," 2011.

[65] RTCA, "Concept of Operations for Aircraft Based Conflict Detection and Resolution," *October*, no. October, 1999.

[66] W. Kelly, "Conflict detection and alerting for separation assurance systems," *Prodeedings of the 18th Digital Avionics Systems Conference*, no. October 1995, 1999.

[67] R. A. Paielli and H. Erzberger, "Conflict Probability Estimation for Free Flight," 1996.

[68] M. Jardin, "Air traffic conflict models," *Proceedings of the Aviation Technology, Integration and Operations Conference (ATIO)*, no. September, pp. 1–13, 2004.

[69] SESAR, "SESAR master plan D5," 2008.

[70] H. Erzberger, "Separation Assurance in the Future Air Traffic System," no. March. 2009.

[71] FAA, "NextGen Implementation Plan March 2012," 2012.

[72] D. of Transportation, "Concept of Operations for the Next Generation Air Transportation System, V2," 2007.

[73] B. Abdul-Baki, "Independent validation and verification of the TCAS II collision avoidance subsystem," *Aerospace and Electronic Systems Magazine*, no. August, pp. 3–9, 2000.

[74] M. J. Kochenderfer, L. P. Espindle, J. K. Kuchar, and J. D. Griffith, "A Comprehensive Aircraft Encounter Model of the National Airspace System," 2008.

[75] M. J. Kochenderfer and J. K. Kuchar, "Uncorrelated Encounter Model of the National Airspace System, Version 1.0," 2008.

[76] M. J. Kochenderfer, L. P. Espindle, J. K. Kuchar, and J. D. Griffith, *Correlated encounter model for cooperative aircraft in the national airspace system version 1.0*. 2008.

[77] J. P. Chryssanthacopoulos and M. J. Kochenderfer, *Collision avoidance system optimization with probabilistic pilot response models*. IEEE, 2011, pp. 2765–2770.

[78] M. W. Edwards, M. J. Kochenderfer, J. K. Kuchar, and L. P. Espindle, "Encounter models for unconventional aircraft version 1.0," 2009.

[79] R. D. Smallwood and E. J. Sondik, "The Optimal Control of Partially Observable Markov Processes Over a Finite Horizon," *Operations Research*, vol. 21, no. 5, pp. 1071–1088, 1973.

[80] L. Winder and J. Kuchar, "Hazard Avoidance Alerting with Markov Decision Process," Massachusetts Institute of Technology, 2004.

[81] R. Weibel, "Establishing a Risk-Based Separation Standard for Unmanned Aircraft Self Separation," *11th AIAA Aviation Technology, Integration, and Operations (ATIO)*, no. September, 2011.

[82] S. Temizer, M. Kochenderfer, and J. K. Kuchar, "Collision avoidance for unmanned aircraft using Markov decision processes," *Proc. AIAA Guidance, Navigation and Control Conference*, pp. 1–22, 2010.

[83] A. Drumm, J. Kuchar, A. Lacher, and A. Zeitlin, "Collision Avoidance for Unmanned Aircraft: Proving the Safety Case.," *Analysis*. pp. 1–12, 2007.

[84] M. J. Kochenderfer, K. J. Shih, J. P. Chryssanthacopoulos, C. E. Rose, and T. R. Elder, *Position validation strategies using partially observable Markov decision processes*. IEEE, 2011, pp. 4A2–1–4A2–9.

[85] R. Chamlou, "Design Principles and Algorithm Development for two types of NextGen Airborne Conflict Detection and Collision Avoidance," *Integrated Communications Navigation and Surveillance*, pp. 1–12, 2010.

[86] R. Chamlou, "NextGen Airborne Conflict Detection and Collision Avoidance System (NextCAS)," no. August. 2010.

[87] RTCA, *DO-319: Safety, Performance and Interoperability Requirements Document for Enhanced Traffic Situational Awareness During Flight Operations (ATSA-AIRB)*. 2010.

[88] S. Silva, "Human Factors Studies of an ADS-B Based Traffic Alerting System for General Aviation," Massachusetts Institute of Technology, 2012.

[89] C. Wickens, S. Rice, and D. Keller, "False Alerts in Air Traffic Control Conflict Alerting System: Is There a 'Cry Wolf' Effect?," *The Journal of Human Factors*, vol. 51, no. 4, pp. 446–462, 2009.

[90] ATSB, "Limitations of the See-and-Avoid Principle," 1991.

[91] L. C. Yang, "Aircraft conflict analysis and real-time conflict probing using probabilistic trajectory modeling," Massachusetts Institute of Technology, 2000.

[92] L. C. Thomas, C. D. Wickens, and E. M. Rantanen, "Imperfect Automation in Aviation Traffic Alerts: A Review of Conflict Detection Algorithms and Their Implications for Human Factors Research," *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, vol. 47, no. 3, pp. 344–348, Oct. 2003.

[93] M. D. Nisar, "Minimax Robustness in Signal Processing for Communications," pp. 11–15, 1960.

[94] G. Li, J. Hu, S.-W. Wang, P. G. Georgopoulos, J. Schoendorf, and H. Rabitz, "Random sampling-high dimensional model representation (RS-HDMR) and orthogonality of its different order component functions.," *The journal of physical chemistry. A*, vol. 110, no. 7, pp. 2474–85, Feb. 2006.

[95] A. Saltelli, M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola, *Global Sensitivity Analysis: The Primer*. John Wiley & Sons Ltd, 2008.

[96] A. Forrester, Andras Sobester, and A. Keane, "Engineering Design via Surrogate Modelling - A Practical Guide." John Wiley & Sons.

[97] SC-186 and RTCA, "DO-289: Minimum Aviation System Performance Standards for Aircraft Surveillance Applications (ASA)." 2003.

[98] ICAO, "Aeronautical Telecommunications, Annex 10," *International Standards and Recommended Practices*, 2001.

[99] T. Ziehn and A. Tomlin, “GUI-HDMR,” 2008. [Online]. Available: http://gui-hdmr.de/.

[100] The Avidyne Corporation, “TSAA Program Flight Test Final Report,” Bedford, MA, 2013.

[101] European Union, *Regulation No 1207/2011: EU ADS-B Out Rule*, no. 1207. 2011, pp. 35–52.

[102] M. Warwick, “Going Live,” *Aviation Week & Space Technology*, pp. 46–47, 2010.

[103] FAA, *Order JO 7110.65T*. Federal Aviation Administration, 2010.

[104] M. R. Jenkins, “Equipage Requirements, Benefits, and Risks of ADS-B Applications by,” Massachusetts Institute of Technology, 2009.

[105] G. E. Kocher, “Task 3.2 Analysis of the past 10 years of National Transportation Safety Board aviation mid-air collision and available near miss data to assess the impact AWSAS might have had on these collision events.” Not Publically Available, 2009.

[106] Administration National Aeronautics and Space, “Aviation Safety Reporting System.” .

Appendix A

OVERVIEW OF THE US ADS-B SYSTEM ARCHITECTURE

As part of the Federal Aviation Administration's (FAA) plans to modernize the Air Traffic Control (ATC) system, ADS-B, supplemented by the current radar system, is the basis of the future surveillance system in the US [4]. Similar plans exist in Europe and other parts of the world [101]. ADS-B takes advantage of the fact that most modern aircraft have advanced navigation systems that use the global navigation satellite system (GNSS) and are often capable of determining the aircraft's position and velocity much more accurately than radar. The aircraft then broadcasts this more accurate information, which thus has the potential to provide higher position and velocity accuracy, direct heading information, and geometric and barometric altitude. Also, at once per second, ADS-B has a higher update rate than radar, which updates once every 4.8 seconds in the terminal area and once every 12 seconds in en-route airspace.

Figure A-1 is a schematic representation of the overall ADS-B system. At least once per second, aircraft equipped with ADS-B avionics broadcast their position, altitude, direction and magnitude of ground speed, and other information pertinent to pilots and air traffic controllers. This broadcast is called "ADS-B Out" and is depicted by the blue arrows in Figure A-1. Ground stations receiving these ADS-B messages forward them via a private network to the FAA facilities responsible for display on the air traffic controller's screen. Other aircraft in the vicinity also can receive ADS-B Out messages. The capability of receiving ADS-B on-board the aircraft is defined as "ADS-B In" (green arrows in Figure A-1).

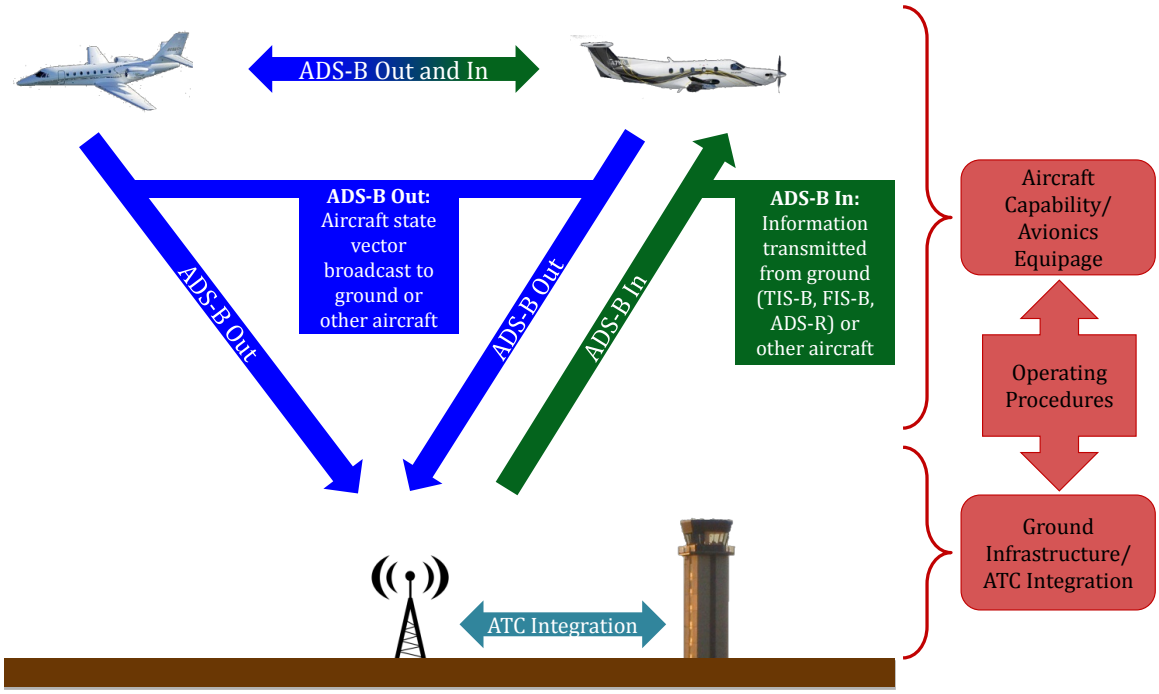


Figure A-1: Schematic Representation of ADS-B

ADS-B In messages can be used to display traffic in the vicinity to the pilot using a cockpit display of traffic information (CDTI, Figure A-2).

Figure A-2: Cockpit Display of Traffic Information (CDTI)

ADS-B also has a data link capability. Messages can originate from the ground stations and be used to uplink additional data directly into the cockpit of appropriately equipped

aircraft. Two types of data link messages have been defined: Traffic Information Service – Broadcast (TIS-B) and Flight Information Service – Broadcast (FIS-B).

FIS-B was originally introduced to increase user benefit to GA and thus increase equipage incentives. However, the frequency originally proposed for ADS-B (1090MHz) had insufficient bandwidth to support FIS-B[17]. As a result, the FAA decided to implement a dual link strategy and provide ADS-B services on two frequencies: 1090ES ADS-B, used mostly for Air Transport, and Universal Access Transceiver (UAT) ADS-B for GA Table A-1 outlines the main differences between the two links. Note that FIS-B is only available on UAT.

Table A-1: Differences Between 1090-ES and UAT ADS-B Link

	Mode S Extended Squitter 1090ES	**Universal Access Transceiver (UAT)**
Frequency	1090 MHz	978 MHz
Frequency Shared With	TCAS, Primary Radar, TIS-B, ADS-R	FIS-B, TIS-B, ADS-R
Intended User	Air Transport, High-End General Aviation	General Aviation
Technical Standard	DO-260B, as outlined in TSO-166b	DO-282B, as outlined in TSO-154c

The decision to implement two separate links introduces additional complexity to the ADS-B system: aircraft on one link are not able to receive ADS-B messages transmitted on the other frequency. To address this issue, Automatic Dependent Surveillance – Rebroadcast (ADS-R) was implemented. ADS-R is the capability of ADS-B ground stations to rebroadcast messages received on the UAT link to the 1090ES link and vice versa. This allows aircraft equipped with ADS-B In to receive ADS-B Out messages from the other link with a one second delay.

Introducing UAT also has implication on an international level. The international ADS-B standard is the 1090ES link; any aircraft with UAT ADS-B avionics would have to follow special procedures to leave the US since it would not comply with the international 1090ES ADS-B standard.

[17] 1090MHz is the interrogation frequency for ground based RADAR. Also, TCAS operates on that same frequency. Concerns exist that adding ADS-B, TIS-B and FIS-B to 1090 would overly congest it and reduce the efficiency of TCAS and RADAR.

The FAA has divided ADS-B services into two criticality levels: “Critical” and “Essential”. ADS-B and ADS-R messages are considered Critical because they support applications such as aircraft surveillance and separation. TIS-B and FIS-B are considered Essential because they are advisory in nature and support applications at an essential but not a critical level [64].

As indicated in Figure A-1, the overall system architecture can be broken down into three major system elements: aircraft capability, ground infrastructure and operating procedures, each of which will be addressed individually.

A.1 AIRCRAFT CAPABILITY – AIRCRAFT AVIONICS

The airborne capability of ADS-B consists of the ADS-B avionics on board appropriately equipped aircraft. In 2009, the FAA published the ADS-B mandate that dictates the required capabilities of these ADS-B avionics. This section introduces the airborne capability and its requirements as part of the overall ADS-B system architecture.

Every ADS-B avionics architecture compliant with the mandate has two core components: a navigation unit providing position and velocity information and an ADS-B transceiver transmitting that information on one of the two link frequencies. One concern among GA is that many active aircraft do not currently have certified navigation units installed. Operators would thus have to equip with a certified navigation unit in addition to an ADS-B transceiver.

A.1.1 ADS-B OUT MANDATE

The ADS-B Out mandate outlines requirements and performance standards for ADS-B Out avionics. The rule states that “… [ADS-B Out] equipment will be required for aircraft operating in classes A, B and C airspace [and] certain class E airspace.” This Class E airspace is airspace above 10,000ft and within the Mode C veils of busy airports. Currently, the FAA is not mandating ADS-B In equipage [5].

The rule also dictates the minimum contents of the ADS-B message and sets performance requirements for each of those elements. These performance requirements were set to enable ATC to conduct aircraft surveillance with ADS-B at a level equivalent to the current radar based system. However, certain proposed applications of ADS-B may require higher performance requirements than those outlined in the rule. Operators who want to use those applications would have to have equipment that meets those higher requirements. Table A-2 lists the required message elements.

Table A-2: Minimum Required ADS-B Message Elements and Their Minimum Performance Requirements

ADS-B Message Element	Performance Requirement	Notes
Length and Width of Aircraft	Hardcoded	Only Transmitted on Ground
Latitude and Longitude	See NACp	In reference to WGS84
Barometric Altitude	N/A	In 25ft Increments
Aircraft Velocity	See NACv	In m/s
TCAS Installed	Hardcoded	Yes or No coding
TCAS RA In Progress Flag	N/A	Yes of No coding
ATC Transponder Code	N/A	Entered via same interface as Transponder
Aircraft Call Sign	N/A	Either N-number or Airline Call Sign
Emergency Status	N/A	Flag to indicate Emergency, Radio Failure or Unlawful Interference
IDENT	N/A	Same function as Transponder IDENT
24-bit ICAO aircraft address	Hardcoded	Binary Code Assigned by ICAO
Emitter Category	Hardcoded	Gives indication of type of aircraft
ADS-B In Equipment	Hardcoded	Yes or No coding
Geometric Altitude	N/A	Height above WGS84
NACp (Navigational Accuracy Category for Position)	Less than 0.05NM (NACp=8)	Minimum Required Position Accuracy
NACv (Navigation Accuracy Category for Velocity)	Less than 10m/s (NACv=1)	Minimum Required Velocity Accuracy
NIC (Navigation Integrity Accuracy)	Less than 0.2NM	Minimum required Integrity
SDA (System Design Assurance Parameter)	Hardcoded, at least 2 (10e-5)	Maximum probability of false or misleading data to be transmitted
SIL (Source Integrity Level)	Hardcoded, at least 3 (10e-7)	Maximum probability of exceeding the NIC containment radius

A.2 ADS-B GROUND INFRASTRUCTURE

The physical ADS-B ground infrastructure consists of the physical ADS-B antennas on the ground, the network infrastructure required to transmit the received messages to the relevant ATC centers as well as the systems required to fuse the surveillance data from ADS-B with surveillance data from the currently existing radar infrastructure.

The FAA externally subcontracted the deployment of the nationwide ADS-B system. Figure A-3 shows the predicted ADS-B coverage for the US at full implementation. Areas highlighted in blue have a predicted ADS-B surveillance coverage at or below 1800 ft AGL.

The US is expected to deploy 794 ADS-B ground stations (Figure A-3) by 2013. The contract requires the ADS-B surveillance volume to be equivalent or larger than the currently existing radar volume. However, given the number and locations of planned stations, the actual ADS-B coverage is expected to exceed radar coverage in many areas.

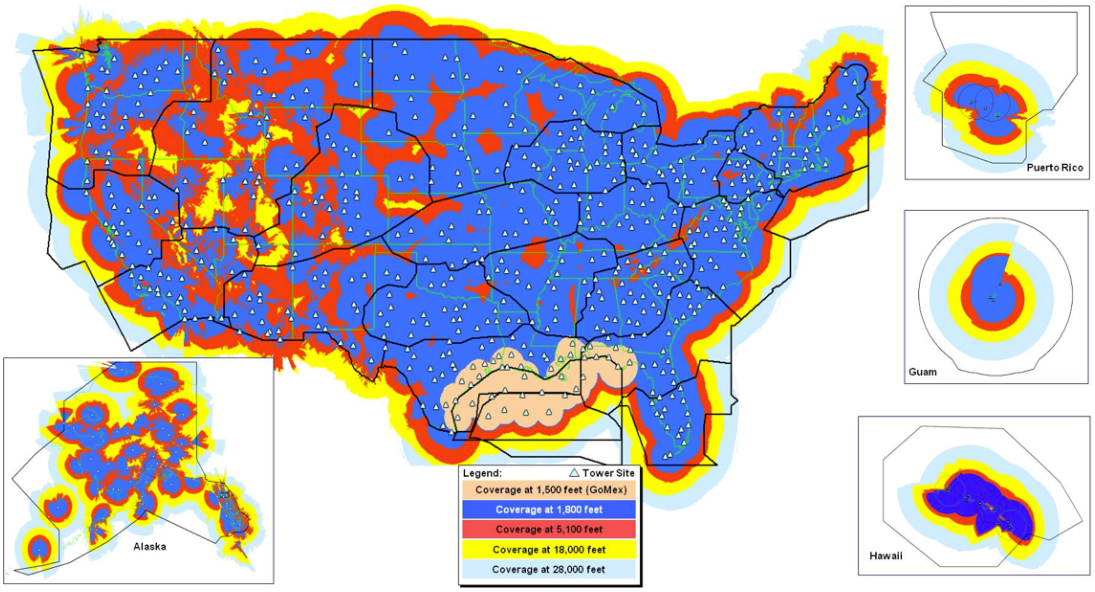


Figure A-3: Predicted ADS-B Coverage at Full Implementation

Some stations will be collocated with existing radar infrastructure. However, most ground stations will be self-contained towers housing one omni-directional UAT antenna and four directional 1090MHz antennas. The towers also have two dual channel communication radios and antennas and in some locations an automatic weather observation station (AWOS) station. To support operations during a loss of electrical power, each station has a diesel generator and batteries.

Figure A-4: Temporary Installation of an ADS-B Antenna on a Terminal Area Radar Tower in Brisbane, Australia (credit: Greg Dunstone)

Once received by the ground station, ADS-B messages are routed via private networks to three control stations in Ashburn, VA, Dallas, TX, and Phoenix, AZ. At those control stations, duplicates, which occur if more than one station received the message, are removed and all messages are grouped by geographical location. "The control stations then validate targets in one of three ways: correlation with radar data, reports from two 1090 radios with the aircraft in view, or pseudo-ranging from a single UAT radio which time tags transmissions. ADS-B messages are then forwarded to the FAA marked as 'valid', 'invalid' or 'unknown'." [102]. This process is completed within 0.7 seconds of receiving the ADS-B message at the ground station. The three control stations also receive the radar data from the nationwide Host Air Traffic Management Data Distribution System (HADDS) and use it to create the TIS-B messages.

A.3 ADS-B Operating Procedures

ADS-B Operating Procedures will supplement the current ATC procedures and outline the interactions between the airborne and the ground based elements of the ADS-B system.

Current radar-based ATC procedures are outlined in FAA/DOT Order 7110.65S, "Air Traffic Control" [103]. This order is a collection of rules describing how air traffic is to be directed in the NAS by air traffic controllers. A majority of those procedures are for regulating flight in Instrument Meteorological Conditions (IMC). In addition to JO 7110.65S, Federal Aviation Regulations (FAR) Parts 91, 121 and 135 outline the rules, rights and procedures of pilots and airlines. Lastly, the Aeronautical Information Manual (AIM) lists recommended procedures for flight operations for pilots.

With the introduction of ADS-B as an additional surveillance source, these procedures will need to be amended and updated incorporate ADS-B operations. The expected changes to the existing procedures can be categorized into two groups: adopting of existing radar procedures where ADS-B surveillance is equivalent to radar surveillance and introducing new ADS-B–specific procedures.

A.3.1 ADOPTION OF EXISTING RADAR-BASED PROCEDURES

Adopting existing radar procedures outlined in 71110.65S allows them to be used with ADS-B as well as radar surveillance. As such, this step grants "Radar Equivalence" to ADS-B for surveillance purposes. Examples of procedures in this category include aircraft vectoring, separation services, and VFR Flight Following. In February 2010, the FAA declared "Initial Operating Capability" of ADS-B for surveillance purposes over the Gulf of Mexico. Since then, additional airspace has been added; it is expected that by 2013 ADS-B based surveillance will be available across the US. The improvement in surveillance data quality due to ADS-B may result in a reduction of the "play" present in current operations. Also, the additional information in ADS-B messages may increase overall controller situation awareness.

One promising aspect of ADS-B being considered equivalent to radar is that it would allow the current surveillance coverage volume to expand to remote or mountainous regions at low cost. Although these improvements in surveillance coverage and quality offer some benefit, alone they might not warrant introducing ADS-B and do not take advantage of much of the information available in the ADS-B message. In order to take advantage of this information, new ADS-B–specific procedures will have to be introduced.

A.3.2 INTRODUCTION OF NEW, ADS-B SPECIFIC PROCEDURES

Introducing new ADS-B–specific procedures enables new capabilities in the NAS. Those capabilities are expected to provide a majority of the benefit from ADS-B [4].

In order to introduce new ADS-B procedures, a rigorous process must be followed to ensure safety and effectiveness. The required steps include but are not limited to developing a concept of operations (ConOps), conducting a full safety analysis (known as operational hazard analysis [OHA]), flight testing, and training pilots and air traffic controllers.

The initial focus ADS-B development was deploying the ground infrastructure; as a result, the development and definition of procedures has received less attention. In order ADS-B to be beneficial, operating procedures are required. Therefore, creating operating procedures is of utmost importance for delivering the user benefit that ultimately creates incentives for equipage.

A.4 ADS-B APPLICATIONS

An "ADS-B Application" is a specific purpose for which ADS-B is used in the NAS. ADS-B applications can be grouped into three categories: data link applications, ADS-B Out applications and ADS-B In applications. 32 proposed applications were identified based on a literature review that included FAA technical documentation such as DO-260 and DO-282, EUROCONTROL's Action Plan 23 (which defines ADS-B implementation strategies for Europe), and the Application Integrated Working Plan (v2) [6]. Additionally, in 2009 Jenkins conducted a thorough review of proposed ADS-B applications [104]. The applications listed in her thesis were included in this review as well. The applications were then categorized based on the required ADS-B functionality (Out, In, Data Link) and duplicates were removed. These categories are discussed in the following sections.

A.4.1 ADS-B OUT APPLICATIONS

ADS-B Out applications are based solely on ADS-B Out transmissions and mostly are limited to ATC surveillance applications. Nonetheless, some proposed procedures do take advantage of ADS-B–specific information and introduce new capabilities based on ADS-B Out. Table A-3 is a list of proposed ADS-B Applications.

Table A-3: List of Proposed ADS-B Out Applications

Application Name:	Concept/Description:
ATC Surveillance in Non-radar Airspace (ADS-B-NRA)	Provide ATC surveillance in non-radar areas such as below current radar coverage or offshore operations areas (e.g., Gulf of Mexico) using current radar procedures. Conceivably, new procedures could be created using surveillance information provided by the ADS-B message.
ADS-B Flight Following	Due to the higher coverage volume and the increased surveillance quality and information available, ATC will be able to better advise pilots of nearby traffic, minimum safe altitude warnings (MSAW), etc.
Improved Search and Rescue	Flight track data serves as an input to search and rescue operations. Having better accuracy of the last know position, a faster update rate, more specific information about the aircraft as well as a bigger coverage area, ADS-B will enable more efficient and more accurate responses to emergency situations.
Company/Online Flight Tracking	Current flight tracking is limited to areas with SSR coverage. ADS-B increases this coverage. Information available in the ADS-B message allows aircraft to be identified more readily. This would, allow. e.g., operators or companies to improve their fleet scheduling.

ATC Surveillance for En-Route Airspace (ADS-B-ACC)	ATC will use ADS-B surveillance information in the same manner as radar surveillance, e.g., to assist aircraft with navigation, to separate aircraft, and to issue safety alerts and traffic advisories. The ADS-B surveillance information will be used to enhance the quality of existing radar-based surveillance information. Conceivably, a 3NM separation standard may be acceptable.
ATC Surveillance in Terminal Areas (ADS-B-TMA)	Current radar surveillance will be enhanced in terminal areas. An example would be airports with single radar coverage. ADS-B information could be used to enhance current ATC procedures or ATC automation systems such as tracking or minimum safe altitude warnings (MSAW).
Airport Surface Surveillance and Routing Service	ADS-B surveillance is provided to air traffic controllers to enhance situational awareness with respect to vehicles (including ground vehicles) operating on the airport surface. ADS-B surveillance may also be provided to ground automation and decision support system to aid in the management of traffic flow on the airport surface. This application may allow ASD-X like environments at non ASD-X airports. Conceivably, a pilot or ATC alerting function could be added to this application.
ATC Automation Integration/Automatic Flight Plan Cancellation	Using information provided by the ADS-B message, some ATC functions could be automated. One such application could be automatic flight plan opening or closing.
ADS-B Enhanced Parallel Approaches/ADS-B PRM	This application applies to two different environments. First, it would enhance parallel approaches at airports, which use a precision runway monitoring radar (PRM). ADS-B may enhance surveillance quality. Second, ADS-B surveillance may allow airports without PRM to have a PRM like environment.
ADS-B Emergency Locator Transmitter (ELT)	The ADS-B message has the capability to transmit a "downed aircraft" message. This could double as an ELT functionality.
Enhanced Tower Situational Awareness in Reduced Visibility	Using ADS-B, a virtual image could be created to aid situation awareness for tower controllers.
ADS-B Enabled Portable Devices for Airport or FBO Employees	Airline employees (e.g., ramp operators) receive ADS-B reports from aircraft in their fleet and use the data to optimize allocation of ground infrastructure, such as gate space and support vehicles.
Weather Reporting to Ground	If aircraft are equipped accordingly, weather specific information could be transmitted via the ADS-B message improving weather briefings to pilots on the ground and to enhance forecasting.

A.4.2 DATA LINK APPLICATIONS:

Data link applications take advantage of ADS-B's ability to link data directly to the cockpit. TIS-B and Flight Information Service – Broadcast FIS-B are examples of this kind of application. These applications are called "Essential Services" for FAA and ATC purposes and are shown in Table A-4.

Table A-4: List Data Link Applications

Application Name:	Concept/Description:
TIS-B	Using secondary radar surveillance data, messages of non-ADS-B traffic are transmitted to the aircraft. TIS-B is not expected to be required once a threshold level of equipage is achieved.
FIS-B	FIS-B messages contain weather data (such as Doppler radar images) as well as NAS status information (NOTAMS, TFRs, etc.) and are updated every 5 minutes.

With TIS-B, traffic information is linked directly from the ground to the cockpit. ITT, the main contractor installing the ground infrastructure for ADS-B describes TIS-B as follows: "The TIS-B service provides active ADS-B users with a low-latency stream of position reports of non-ADS-B equipped aircraft" (ITT, 2010) These reports are generated using secondary radar data. TIS-B traffic information is added to the ADS-B messages received directly from other ADS-B aircraft via ADS-B In.

TIS-B is not transmitted continuously. A ground station starting to transmit TIS-B to a given aircraft requires two things. First, the aircraft has to be transmitting ADS-B Out and be capable of receiving ADS-B In. Second, there has to be a non-ADS-B target within the vicinity of that aircraft.

The FIS-B service is a broadcast of weather and NAS status information. The broadcast data is specific to the location of a given ground station. FIS-B is broadcast on UAT only. Unlike TIS-B, FIS-B is broadcast regardless of whether any "client" aircraft are in the service volume. FIS-B currently contains the following weather and NAS products [64]:

1. AIRMET
2. SIGMET
3. Convective SIGMET
4. METAR
5. PIREP
6. TAF

7. Winds/Temperatures Aloft
8. CONUS NEXRAD
9. Regional NEXRAD
10. NOTAM
11. SUA

Similar to TIS-B, the information received via FIS-B can be displayed in the cockpit on a separate Multifunction Display (MFD, Figure A-5) or possibly on a CDTI in combination with TIS-B.

Data Link applications are expected to provide substantial benefit to GA. GA often does have access to this kind of data in flight. Providing free access to traffic, weather, and NAS status information is expected to aid flight crews in decision-making and thus reduce accidents.

Figure A-5: FIS-B Information Displayed on MFD

A.4.3 ADS-B IN APPLICATIONS

ADS-B In applications are enabled by the aircraft's ability to receive ADS-B messages. Applications of this kind are expected to introduce new capabilities into the NAS as well as move some of the functions ordinarily performed by ATC to the pilot. Much ADS-B user benefit is expected from this kind of application.

In a recent effort to achieve consensus on the definitions and functionalities of ADS-B In applications, the FAA created the ADS-B Integrated Working Plan (AIWP). A government/industry panel focused on identifying and defining ADS-B In applications wrote the AIWP. Table A-5 lists the applications identified by the AIWP and their descriptions [6].

Table A-5: List of ADS-B In Applications Proposed in the AIWP

Application Name:	Concept/Description:
Traffic Situation Awareness–Basic	Flight crews use this application […] to supplement their visual scan. The display enables detection of traffic by the flight crew. The information provided on the display also reduces the need for repeated air traffic advisories and is expected to increase operational efficiencies.
Traffic Situation Awareness for Visual Approach	The flight crew uses the display to assist in the visual acquisition of a specific target to follow and manual selection of the traffic for coupling. The cockpit display provides ground speed or closure rate information relative to the coupled target continuously throughout the approach.
Airport Traffic Situation Awareness	The application is expected to be used by the flight crew to aid in detection of traffic related safety hazards on taxiways and runways including aircraft on final approach. This assists the flight crew with early detection of traffic conflicts and runway incursions.
Airport Traffic Situation Awareness with Indications and Alerts	Adds alerts and indications to the basic Airport Traffic Situation Awareness application by graphically highlighting traffic or runways on the airport map to inform flight crew of detected conditions, which may require their attention.
Oceanic In-Trail Procedures	Oceanic In-Trail Procedures (ITP) enables flight level change maneuvers that are otherwise not possible within Oceanic procedural separation standards. ITP allows ATC to approve these flight level change requests between properly equipped aircraft using reduced procedural separation minima during the maneuver.
Flight-Deck Based Interval Management–Spacing	Flight-Deck Based Interval Management-Spacing (FIM-S) is a suite of functional capabilities that can be combined to produce operational applications to achieve or maintain an interval or spacing from a target aircraft.
Traffic Situation Awareness with Alerts (TSAA)	Provides pilots and flight crew of non-TCAS equipped aircraft with enhanced traffic situational awareness in all classes and domains of airspace by delivering traffic advisory alerts in the near term.
Flight-Deck Based Interval Management-with Delegated Separation	Flight-Deck Based Interval Management-Delegated Separation (FIM-DS) is a suite of functional capabilities that build upon FIM-S and can be combined to produce operational applications that delegate responsibility for separation from a target aircraft to the flight crew.
Independent Closely Spaced Routes	This airborne capability is expected to facilitate closer spacing between routes, which will enable greater use of terminal, en route, and oceanic airspace.
Paired Closely Spaced Parallel Approaches	To allow flight crews to conduct instrument approach procedures simultaneously to closely – spaced parallel runways increasing airport capacity and efficiency of ATC and flight operations.

Independent Closely Spaced Parallel Approaches	When weather conditions dictate the use of instrument approaches, arrival rates decrease, resulting in delays. It is expected that Independent Closely Spaced Parallel Approaches (ICSPA) will be applicable to runways spaced between 2,500 and 4,300 feet.
Delegated Separation–Crossing	Enables ATC to resolve a conflict by issuing either a lateral or vertical crossing clearance and delegating separation responsibility to the flight crew with respect to ATC designated target aircraft.
Delegated Separation–Passing	Enables ATC to resolve an along-track overtake conflict by issuing either a lateral or vertical passing clearance and delegating separation responsibility to the flight crew with respect to an ATC designated target aircraft.
Flight Deck Interval Management – Delegated Separation with Wake Risk Management	Increases capacity by enabling reduced airborne separation minima within the current wake avoidance limits by providing aircraft-based tools for managing wake risk when conducting delegation separation with FIM-DS.
ADS-B Integrated Collision Avoidance	Further increases capacity by enabling reduced airborne separation minima. This is achieved by integrating ADS-B data with the TCAS system to create a more robust collision avoidance system (CAS) for ground separation, delegated separation, and self-separation operations in all conditions.
Flow Corridors	Flow corridors consist of tubes or "bundles" of near-parallel trajectories in the same direction, which consequently achieve a very high traffic throughput, while allowing traffic to shift as necessary to enable more effective weather avoidance, reduce congestion, and meet special use airspace (SUA) requirements.
Self-Separation	The flight crew of a self-separating aircraft assumes responsibility from the ATC for separation from all traffic for a defined segment of the flight. As part of its delegated separation responsibility, the flight crew is granted authority to modify its trajectory within defined degrees of freedom without renegotiating with ATC.

Appendix B

IDENTIFYING HISTORICAL INTERACTIONS BETWEEN COLLISION ALERTING SYSTEMS AND COLLISIONS AVOIDANCE SYSTEMS

As mentioned in Chapter 3, an encounter between an aircraft equipped with a collision *alerting* system and an aircraft with a collision *avoidance* system is of particular interest. This type of encounter is possible with current alerting systems. Therefore, the accidents reviewed in section 3.2 were evaluated again to identify any case in which the presence of an alerting system may have contributed to the accident. Additionally, a study conducted by the Navy is also summarized here.

Of the 112 NTSB accident reports, only 4 mentioned the presence of a traffic alerting system on one or both aircraft. Those 4 accidents are summarized in Table B-1. Based on the types of aircraft involved in two other reported accidents, it is possible that a collision alerting system was available on one or both aircraft but was not mentioned in the report. Those two accidents were also included in Table B-1.

Table B-1: Summary of NTSB Reports in Which at Least One Aircraft Had a Traffic Alerting System (* These accidents did not mention traffic alerting systems in the reports but may have had one)

NTSB Report #	Aircraft 1	Aircraft 2
LAX04FA095A	Beech 95-B55, TCAD model 9900BX	Cessna 180K Transponder
LAX01FA018B	Gulfstream G-1159A, TCAS II	Beech C90 Transponder
LAX06FA277B	Hawker 800XP, TCAS II	Glider, Mode C Transponder, turned off
DEN08MA116A	Bell 407, SkyWatch	Bell 407, Transponder
20001212X22313*	F-16	Cessna 172 Transponder
20001212X21286*	LearJet	Extra 300 Transponder

As Table B-1 shows, there were no mid-air collisions where both aircraft were operating a traffic alerting system. Specifically, mid-air collisions that involved an aircraft equipped with a TCAS II system only involved other aircraft not equipped with a traffic alerting system.

The ASRS reports for the same 10-year period used in the NTSB review were reviewed again. The database was searched for any report categorized as an NMAC and contained any of the following words:

- TAS
- Caution
- Alert
- SkyWatch
- FLARM
- GDL 90

The 256 resulting reports were narrowed down further by searching for the following encounter characteristics:

Aircraft 1 (reporting aircraft):	Non-Part 121 or Part 135, assumed to not have TCAS II
Aircraft 2:	Part 121, assumed to have TCAS II

Two reports were of interest. The abbreviated descriptions of what the reporting pilots observed are shown below:

Report 1 (ACN# 921439):

> "We were level at 11,000 southeast bound on an IFR flight plan crossing the Orlando Class B airspace. We were being vectored 'Overtop MCO.' A B737 was climbing out of MCO, southwest bound climbing off the runway. I was asked if I had him in sight, 'I would like to keep him climbing.' I stated I had the traffic insight. [...] The Controller told the B737 to climb and maintain 16,000 then said, 'the B737 will be crossing your altitude.' I never said I would maintain visual separation, I just said I had him insight. Next thing I knew, they were a collision factor. Our TAS system sounded and I told the Controller, 'I don't think this will work.' Controller said turn left (into the path of the B737) I started to roll left, but rolled the wings level. Next the B737 crew stated they had an 'RA' and took action, Crossing 300-400 FT below us. [...]"

Report 2 (ACN# 519955):

> "At 2000ft in cruise flight approx. 12 nm prior to the HTO VOR, I observed an aircraft on the SkyWatch traffic system descending to my altitude. I switched to the 2 nm range and observed no bearing change relative to my aircraft. At 1½ NM and 200ft vertical separation I queried New York Approach (132.25) as to the intentions of the 'challenger.' ATC told the challenger n-number to immediately climb. At 1/2 nm and 100ft separation on SkyWatch, I initiated emergency evasive descent maneuver to 1500ft MSL and observed the challenger come from behind a cloud directly overhead. I then notified ATC that I had deviated from my cleared altitude and was climbing back to 2000 ft. […]"
>
> Supplemental info from ACN# 518975: "TCAS II showed him within 100ft vertically and less than 2mi -- the last time I looked at the inst. The only way we avoided the collision was our captain shoved the nose over and we descended 500 ft."

In the first report, the pilot's initial action was due to ATC instruction and not due to input from an alerting system. In the second report, the pilot's action was based on input from the SkyWatch system; however, that action resolved rather than aggravated the situation. In summary, the reviewed NTSB and ASRS reports did not contain any cases of an alerting system's presence negatively affecting the outcome of an airborne encounter with a TCAS II–equipped aircraft.

In 2009, the US Navy conducted a study to evaluate the collision alerting systems' potential impact on mid-air collisions [105]. This study evaluated 5 specific accidents in depth in order to determine the potential benefit and usefulness of using TCAS II in small aircraft. Figure B-1 shows a summary for each of the reviewed accidents.

These accidents agree with Table B-1 with the exception of the Hoboken, NJ and the Phoenix AZ mid-air collisions. The NTSB report for the Hoboken accident was filed after the evaluation period for Study 1 mentioned above. The NTSB report for the Phoenix, AZ accident did not explicitly mention a traffic alerting system.

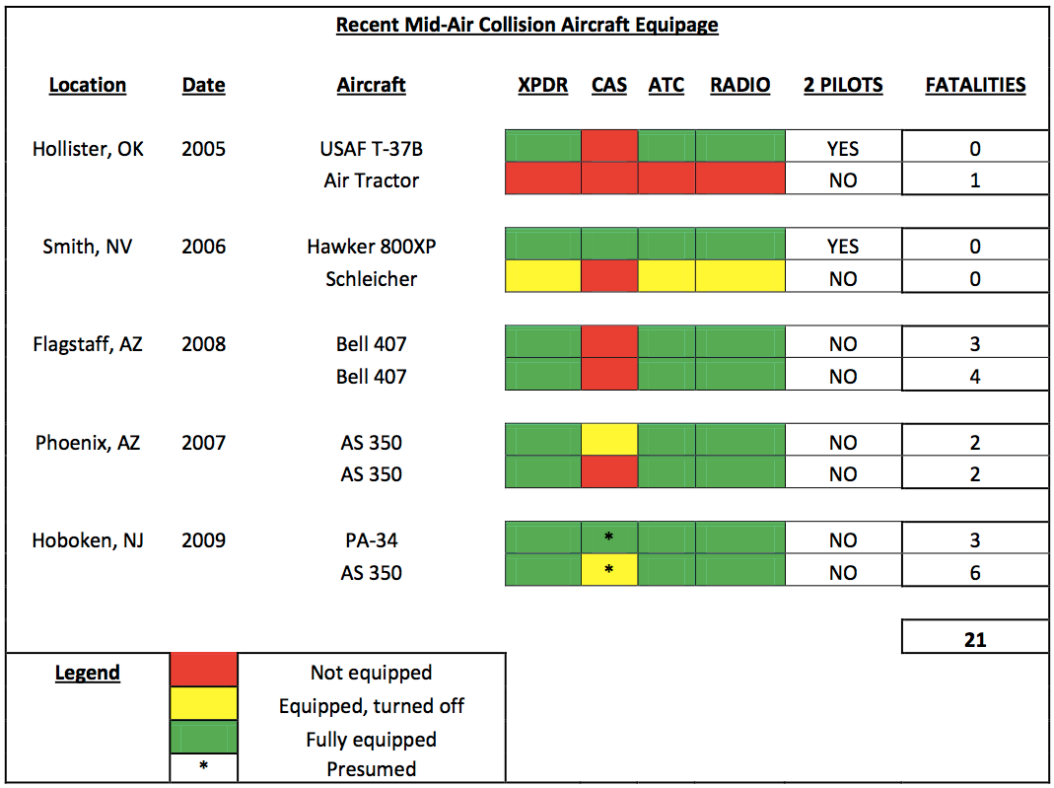


Figure B-1: Accidents Evaluated by the Navy Study

As Figure B-1 shows, even though alerting systems may have been available on both aircraft, no mid-air collisions occurred when both flight crews were using a traffic alerting or TCAS II system. This observation confirms the conclusions derived from the NTSB accident analysis that there are no reported cases in which the presence of an alerting system negatively affected the outcome of an airborne encounter with an aircraft equipped with TCAS II.

Appendix C
MATLAB CODE OF SAMPLE TSAA ALGORITHM

(Intentionally left blank)

Contents

Main TSAA Code

```
function threat_aircraft = TSAA(mode, varargin)
```

Introduction

This function is the exemplar MATLAB implementation of the TSAA algorithm developed under the FAA/MIT TSAA Project. For more infromation, contact Fabrice Kunzi at [kunzi at alumn.mit.edu]

This function was wirtten such that it can be called from the sample tracker in DO-317A. It assumes that the data of all targets is passed to it in a common, carteisan frame of reference, extrapolated to a common point in time.

Inputs

When TSAA is called, it can be called in three modes: 'update', 'detect' and 'return'.

mode: Either 'update', 'detect' or 'return' varargin: Only required when TSAA is called in the 'update' mode.

If TSAA is called in 'update' mode, varargin must be a structure containing at least the following fields (sample values shown):

```
     t: 41312       Time stamp of the current update,
                    in seconds (extrapolated)
   num: 4           Target ID; the ownship is assumed to
                    be num = 1
     x: -7876.6     x Position (m), extrapolated to time specified
                    by the t field
     y: 8815.5      y Position (m), extrapolated to time specified
                    by the t field
     z: 544.55      z Position (m), extrapolated to time specified
                    by the t field
  xdot: -15.801     x (North) velocity (m/s)
  ydot: 46.503      y (East) velocity (m/s)
```

```
zdot: -0.30375    z (vertical) velocity (m/s)
 tor: 41309       Time at which the last report was
                  received for this target ('time of reception')
```

Outputs

TSAA returns a single variable ('threat_aircraft') which is a structure that contains all aircraft that are currently in alert state. Each aircraft in the array is itself a structure named after the aircraft's ID concatenated with 'target_'. Each target's structure contains a binary 'PAZ' and 'CAZ' field that indicate alert state. For example, a target with the ID '2' that is currently in a PAZ alert state would look as follows:

```
threat_aircraft.target_2.PAZ = 1
threat_aircraft.target_2.CAZ = 0
```

Revisions

05/17/2013: Version 1.0 - Fabrice Kunzi

06/17/2013: Version 2.0 - Fabrice Kunzi Added additional logic to prevent alerts on separating targets, wrap-around alerts and to maintain alerts on targets that continue to close but don't have predicted conflicts.

Variable Definitions

To ensure that TSAA has access to the historical track of a particluar target, the following variables are defined as persistent variables (which will prevent them from being erased each time the TSAA function call completes). The description of each variable is given below:

```
persistent config tsaa_ownship tsaa_targets

if isempty(config)
    % A set of adjustable, algorithm internal parameters that control
    % the behavior of the algorithm:
    config = tsaa_config();

    % An instance of the tsaa_aircraft_class that stores the state data
    % for the ownship as it becomes avaiabale. See documentation in
    % tsaa_aircraft_class.m
    tsaa_ownship = tsaa_aircraft_class('Ownship');

    % An array that contains instances of the tsaa_aircraft_class for
    % each target that is actively monitored by TSAA.
    tsaa_targets = [];
end

% The array that is passed back to the ASSAP processor that contains
% all aircraft that are either in PAZ or CAZ alert state. Refer to the
% "Outputs" section above for more detail. This array is initialized
% each time TSAA is called.
threat_aircraft = struct();

% This last variable determines whether TSAA is run in diagnostics
% mode. If set to 1, the Alert_Logging.m script is called and the the
% alerts issued by TSAA are stored for later use. If set to 2, the
% Alert_Logging.m and the TSAA_Analysis_Tools.m scripts are called
% every time TSAA runs in 'detect' mode.
diagnostics_output = 2;
```

TSAA Functions

The rest of this script performs the actual target maintenance and threat detection.

```
    switch mode
```

'Update' Case

The 'update' mode will store the state data for the aircraft for which new information has become available:

```
        case 'update'
```

```
            data = varargin{1};

            % Check to see if the data is for the ownship:
            if data.num == 1
                tsaa_ownship = tsaa_ownship.update(data);
            else
            % If not, check to see if the update belongs to a target that
            % is already being tracked:
                tracked = 0;
                for idx = 1:numel(tsaa_targets)
                    if data.num == tsaa_targets(idx).num
                        tsaa_targets(idx) = tsaa_targets(idx).update(data);
                        tracked = 1;
                    end
                end

                % If the target is not currently being tracked, generate a
                % new instance of the tsaa_aircraft_class for the new
                % target:
                if tracked == 0
                    tsaa_targets = [tsaa_targets ...
                                    tsaa_aircraft_class(data)];
                end

            end
```

'Detect' Case

The 'detect' mode performs a pariwise evaluation of each target in the tsaa_targets structure to determine whether it is predicted to be a threat to the own-ship:

```
        case 'detect'
```

```
            % Propagate the trajectory for the ownship:
            tsaa_ownship = tsaa_ownship.propagate_trajectory(config);
```

```
% As TSAA steps through the list of targets, it takes note of
% the targets that have become stale. Those targets are
% removed from the tsaa_targets structure at the end of the
% 'detect' case.
bad_targets = [];

% Step through all the known targets, propagate their
% trajectories and evaluate whether they pose a threat to the
% ownship:
for idx = 1:numel(tsaa_targets)
    target = tsaa_targets(idx);

    % Check to see if the time when the last ADS-B report was
    % received is within the maximum allowable time for a
    % target to be a TSAA target:
    if  target.tracker_t(end) - target.report_t(end) > ...
        config.maxDataAge

        bad_targets = [bad_targets idx];

        % In case that the TSAA_Output structure is to be
        % generated, Alert_Logging is called:
        if ~isempty(target.t_PAZ_Alerts_On) && ...
           diagnostics_output ~= 0

            [~] = Alert_Logging(target);

        end
        continue

    % Next, check to see if the current target has enough data
    % to actually qualify:
    elseif length(target.tracker_t) < ...
           config.numberOfHitsToQualify
        continue
    end

    % Generate the trajectory for the current target (Note,
    % this occurs in the aircraft object).
    target = target.propagate_trajectory(config);

    % Perform the conflict detection between the current target
    % and the ownship:
    target = detect_conflicts(target, tsaa_ownship, config);

    % Based on the predicted conflicts, determine whether an
    % alert is necessary:
    target = alerting_logic(target, tsaa_ownship, config);

    % If a the current target is in alert state, it is stored
    % in the threat_aricraft sturcture to be returned to the
    % ASSAP processor.
    if target.PAZ_Alert_Status(end) == 1
        name = strcat(['target_', num2str(target.num(end))]);
        threat_aircraft.(name) = ...
            struct('PAZ', target.PAZ_Alert_Status(end),...
                   'CAZ', target.CAZ_Alert_Status(end));
```

```
                end

                % Store the now updated target object in the tsaa_targets
                % array:
                tsaa_targets(idx) = target;
            end

            % As a last step of the threat detection, the targets that
            % expired and were identified as such during this call are
            % removed from the tsaa_targets array.
            tsaa_targets(bad_targets) = [];

            % To be used for analysis and diagnostics: Any additional
            % functionality that may be desired can be placed in the
            % TSAA_Analysis_Tools.m script which will be called every time
            % TSAA is called in the 'detect' mode.
            if diagnostics_output == 2
                TSAA_Analysis_Tools(tsaa_ownship, tsaa_targets, config)
            end
```

'Return' Case

If TSAA is called in the 'return' mode, the alert_log generated by the Alert_Logging.m script is returned by the TSAA.m script:

```
        case 'return'
```

```
            if diagnostics_output > 0
                for idx = 1:length(tsaa_targets)
                    threat_aircraft = Alert_Logging(tsaa_targets(idx));
                end
                % Add the algorithm configuration to the output structure:
                threat_aircraft.algo_config = tsaa_config();
            end

            return
    end
```

```
Error using TSAA (line 101)
Not enough input arguments.
```

```
end
```

Definition of Algorithm Internal Parameters

This function returns the algoritm internal parameters that control the behavior of the TSAA algorithm.

```
function config = tsaa_config()
    ft2m = 0.3048;
```

```
    % TSAA Parameters Version 5
    config = struct('rcaz', 500 * ft2m, ...
                    'vcaz', 200 * ft2m, ...
                    'rpazmin', 750 * ft2m, ...
                    'vpazmin', 450 * ft2m, ...
                    'tauPAZhor', 8, ...
                    'tauPAZvert', 2, ...
                    'lookahead', 35, ...
                    'dt', 1, ...
                    'doubleTrigger', 15, ...
                    'reAlertDelay', 6, ...
                    'numberOfHitsToQualify', 2, ...
                    'numberOfUpdatesTurn', 5, ...
                    'numberOfUpdatesVert', 11, ...
                    'maxDataAge', 15, ...
                    'closureThreshold', 0);

end
```

Threat Detection

This function calculates the values needed to determine whether the current target is in conflict with the ownship. To do so, it uses the previousley calculated and stored trajectorys in the target and ownship objects. Each column added to the target's trajectory is described individually. Note: The data calculated in this function is only stored in the target object's trajectory array.

```
function target = detect_conflicts(target, ownship, config)
    % The eigth column is the separation between the two aircraft in the
    % x-dimension:
    target.traj(:, 8) = target.traj(:,2) - ownship.traj(:,2);

    % The ninth column is the separation between the two aircraft in the
    % y-dimension:
    target.traj(:, 9) = target.traj(:,3) - ownship.traj(:,3);

    % The tenth column in the target's trajecotry array is the
    % horizontal range between the two aircraft (Note: this should never be
    % zero as it will cause a ZeroDivisionError later.)
    target.traj(:, 10) = (target.traj(:,8).^2+target.traj(:,9).^2 ).^(1/2);

    % The eleventh column is the relative x velocity (positive is closing):
    target.traj(:, 11) = target.traj(:,5) - ownship.traj(:,5);

    % The twelfth column is the the relative y velocity (positive is
    % closing):
    target.traj(:, 12) = target.traj(:,6) - ownship.traj(:,6);

    % The thirteenth column is the closure rate between the two aircraft in
    % the horizontal plane (positive is closing):
    target.traj(:, 13) = -(target.traj(:, 8).*target.traj(:,11) +...
                           target.traj(:, 9).*target.traj(:,12))./...
                           target.traj(:, 10);

    % The fourteenth column is the separation between the two aircraft in
    % the z-dimension:
    target.traj(:, 14) = abs(target.traj(:,4) - ownship.traj(:,4));
```

```
    % The fivetheenth column is the relative z velocity. The vertical
    % relative velocity is assumed constant along the trajectory. It's
    % calculation must take the current encounter geomenty into account
    % (again, positive is closing):

    % Along the trajecotry...
    for idx = 1:length(target.traj(:, 1))
        % ... if the ownship is above the target, ...
        if ownship.traj(idx, 4) > target.traj(idx, 4)
            % ... the vertical velocity is:
            target.traj(idx, 15) = target.traj(idx, 7) - ...
                                   ownship.traj(idx, 7);
        % Else, if the target is above the ownship, ...
        elseif target.traj(idx, 4) > ownship.traj(idx, 4)
            % ... the vertical velocity is:
            target.traj(idx, 15) = ownship.traj(idx, 7) - ...
                                   target.traj(idx, 7);
        % Else, if they are co-altitude, ...
        elseif ownship.traj(idx, 4) == target.traj(idx, 4)
            % ... the relative vertical rate is assumed to be 0.
            target.traj(idx, 15) = 0;
        end
    end

    % The sixteenth column is the horizontal PAZ size:
    target.traj(:, 16) = max(config.rpazmin, config.rpazmin + ...
                         config.tauPAZhor .* target.traj(:, 13));

    % The seventeenth column is the vertical PAZ size:
    target.traj(:, 17) = max(config.vpazmin, config.vpazmin + ...
                         config.tauPAZvert .* target.traj(:, 15));

    % The eighteenth column identifies where a PAZ penetration is
    % predicted:
    target.traj(:, 18) = (target.traj(:, 10) < target.traj(:, 16) & ...
                          target.traj(:, 14) < target.traj(:, 17));

    % The nineteenth column identifies where a CAZ penetration is
    % predicted:
    target.traj(:, 19) = (target.traj(:, 10) <= config.rcaz & ...
                          target.traj(:, 14) <= config.vcaz);

end
```

Alerting Logic

The alerting logic uses the predicted conflcits and determines whether an alert is to be issued to the flight crew or whether an ongoing alert is to be maintainted even if no conlficts are predicted. The alert logic consists of 10 subsections:

1. Check if immediate PAZ Alert is necessary 2. Check if a PAZ Alert is necessary due to two consecute confilict predictions more than DoubleTrigger secodns into the future 3. Check if immediate CAZ Alert is necessary 4. Check if a CAZ Alert is necessary due to two consecute confilict predictions more than DoubleTrigger secodns into the future 5. Prevent alerts on targets that are currently separating 6. Prevent alerts on targets with high closure taus (time to CPA with current closure) 7. Prevent alerts from turing off if no conflicts are predicted but the aircraft are still closing on each other 8. Prevent a PAZ alert from turning off while the audio message is being pronounced. 9. Prevent a CAZ alert from turning off while the audio message

is being pronounced.

```
%10. Prevent a CAZ alert from being issued while a PAZ alert is being
%    pronounced.

function target = alerting_logic(target, ownship, config)
    % Determine the time into the future at which the first conflict is
    % predicted:
    traj_time_first_PAZ=target.traj(find(target.traj(:, 18), 1,'first'),1);
    traj_time_first_CAZ=target.traj(find(target.traj(:, 19), 1,'first'),1);


    % ----------- 1. Check if immediate PAZ Alert is necessary: -----------
    % If we predict the violation of the PAZ less than 'DoubleTrigger'
    % seconds into the future, an alert is issued immediately
    if ~size(traj_time_first_PAZ) == 0
        if traj_time_first_PAZ <= config.doubleTrigger
            target.PAZ_Alert_Status = [target.PAZ_Alert_Status 1];

            target.t_to_PAZ_conflict = [target.t_to_PAZ_conflict ...
                                        traj_time_first_PAZ];


        % - 2. Check if a PAZ Alert is necessary due to two predictions: --
        % If we predict the violation of the PAZ more than 'DoubleTrigger'
        % seconds into the future, the conflict has to be predicted by two
        % consecutive TSAA calls:
        elseif (traj_time_first_PAZ > config.doubleTrigger) && ...
               ~(isempty(target.t_to_PAZ_conflict) || ...
                 isnan(target.t_to_PAZ_conflict(end)))

            target.PAZ_Alert_Status = [target.PAZ_Alert_Status 1];
            target.t_to_PAZ_conflict = [target.t_to_PAZ_conflict ...
                                        traj_time_first_PAZ];

        % If a conflict is predicted for the first time and it is further
        % into the future than 'DoubleTrigger' into the future, the alert
        % status remains 0 but the traj_time_first_PAZ value is stored:
        else
            target.PAZ_Alert_Status = [target.PAZ_Alert_Status 0];
            target.t_to_PAZ_conflict = [target.t_to_PAZ_conflict ...
                                        traj_time_first_PAZ];
        end

    % If there is no PAZ conflict predicted along the trajecotry, the alert
    % status remains 0 and a NaN is stored for the t_to_PAZ_conflict:
    else
        target.PAZ_Alert_Status = [target.PAZ_Alert_Status 0];
        target.t_to_PAZ_conflict = [target.t_to_PAZ_conflict NaN];
    end


    % ----------- 3. Check if immediate CAZ Alert is necessary: -----------
    % If we predict the violation of the CAZ less than 'DoubleTrigger'
    % seconds into the future, an alert is issued immediately
    if ~size(traj_time_first_CAZ) == 0
```

```
        if traj_time_first_CAZ <= config.doubleTrigger
            target.CAZ_Alert_Status = [target.CAZ_Alert_Status 1];
            target.t_to_CAZ_conflict = [target.t_to_CAZ_conflict ...
                                        traj_time_first_CAZ];


        % - 4. Check if a CAZ Alert is necessary due to two predictions: --
        % If we predict the violation of the CAZ more than 'DoubleTrigger'
        % seconds into the future, the conflict has to be predicted by two
        % consecutive TSAA calls:
        elseif (traj_time_first_CAZ > config.doubleTrigger) && ...
               ~(isempty(target.t_to_CAZ_conflict) || ...
                 isnan(target.t_to_CAZ_conflict(end)))

            target.CAZ_Alert_Status = [target.CAZ_Alert_Status, 1];
            target.t_to_CAZ_conflict = [target.t_to_CAZ_conflict ...
                                        traj_time_first_CAZ];

        % If a conflict is predicted for the first time and it is further
        % into the future than 'DoubleTrigger' into the future, the alert
        % status remains 0 but the traj_time_first_CAZ value is stored:
        else
            target.CAZ_Alert_Status = [target.CAZ_Alert_Status 0];
            target.t_to_CAZ_conflict = [target.t_to_CAZ_conflict ...
                                        traj_time_first_CAZ];
        end

    % If there is no CAZ conflict predicted along the trajecotry, the alert
    % status remains 0 and a NaN is stored for the t_to_CAZ_conflict:
    else
        target.CAZ_Alert_Status = [target.CAZ_Alert_Status, 0];
        target.t_to_CAZ_conflict = [target.t_to_CAZ_conflict NaN];
    end


    % ------------------ 5. Check for positive closure: ------------------
    % If we predict an alert, check to see if we're currently closing on
    % that target. If the two aircraft are moving apart from each other,
    % don't alert.
    % There are two exception to this rule:
    % 1. The alert is predicted due to a violation of the PAZ at
    % trajectory_t = 0
    % 2. If we observe negative closure but are currently violating the PAZ
    % in the dimension where the negative closure is observed.
    % If (1) or (2) are observed, the alert is issued no matter what the
    % closure rate.

    if (target.PAZ_Alert_Status(end) == 1 && ...
        target.PAZ_Alert_Status(end-1) == 0) && ...
        target.traj(1,18) ~= 1

        % Check the horizontal and vertical closure and the respective
        % separation:
        if (target.traj(1,13) < config.closureThreshold && ...
            target.traj(1, 10) > target.traj(1, 16)) || ...
           (target.traj(1,15) < config.closureThreshold && ...
            target.traj(1, 14) > target.traj(1, 17))
```

```
            target.PAZ_Alert_Status(end) = 0;
        end
    end

    % Repeat for CAZ alerts:
    if (target.CAZ_Alert_Status(end) == 1 && ...
        target.CAZ_Alert_Status(end-1) == 0) && ...
        target.traj(1,19) ~= 1

        % Check the horizontal and vertical closure and the respective
        % separation:
        if (target.traj(1,13) < config.closureThreshold && ...
            target.traj(1, 10) > config.rcaz) || ...
           (target.traj(1,15) < config.closureThreshold && ...
            target.traj(1, 14) > config.vcaz)

            target.CAZ_Alert_Status(end) = 0;
        end
    end


    % ------------------ 6. Prevent Wrap-Around Alerts: ------------------
    % Don't let an alert be issued if the currently observed states do not
    % indicate that the closest point of approach will be reached within
    % the time that TSAA predicts into the future (given by the lookahead
    % time plus the horizontal PAZ scaling).
    %
    % Again, there are two exception to this rule:
    % 1. The alert is predicted due to a violation of the PAZ at
    % trajectory_t = 0
    % 2. If we observe negative closure but are currently violating the PAZ
    % in the dimension where the negative closure is observed.
    % If (1) or (2) are observed, the alert is issued no matter what the
    % closure rate.

    if (target.PAZ_Alert_Status(end) == 1 && ...
        target.PAZ_Alert_Status(end-1) == 0) && ...
        target.traj(1,18) ~= 1

        % Check the closure and vertical taus and the respective
        % separation:
        if (target.traj(1,10)/target.traj(1,13) > ...
            config.lookahead + config.tauPAZhor && ...
            target.traj(1, 10) > target.traj(1, 16)) || ...
           (target.traj(1,14)/target.traj(1,15) > ...
            config.lookahead + config.tauPAZvert && ...
            target.traj(1, 14) > target.traj(1, 17))

            target.PAZ_Alert_Status(end) = 0;
        end
    end

    % Repeat for CAZ alerts:
    if (target.CAZ_Alert_Status(end) == 1 && ...
        target.CAZ_Alert_Status(end-1) == 0) && ...
        target.traj(1,19) ~= 1
```

```
    % Check the closure and vertical taus and the respective
    % separation:
    if (target.traj(1,10)/target.traj(1,13) > ...
        config.lookahead + config.tauPAZhor && ...
        target.traj(1, 10) > config.rcaz) || ...
       (target.traj(1,14)/target.traj(1,15) > ...
        config.lookahead + config.tauPAZvert && ...
        target.traj(1, 14) > config.vcaz)

          target.CAZ_Alert_Status(end) = 0;
    end
end


% ------------------ 7. Check for positive closure: ------------------
% If we were are trying to turn an alert off check to see if the two
% aircraft are physically separating. If not, retain the alert.

% Check to see if we're trying to turn off an alert:
if (target.PAZ_Alert_Status(end) == 0 && ...
    target.PAZ_Alert_Status(end-1) == 1)

    % Check the horizontal and vertical closure:
    if target.traj(1,13) > config.closureThreshold || ...
       target.traj(1,15) > config.closureThreshold

          target.PAZ_Alert_Status(end) = 1;
    end
end

% Repeat for CAZ lerts:
if (target.CAZ_Alert_Status(end) == 0 && ...
    target.CAZ_Alert_Status(end-1) == 1)

    % Check the horizontal and vertical closure:
    if target.traj(1,13) > config.closureThreshold || ...
       target.traj(1,15) > config.closureThreshold

          target.CAZ_Alert_Status(end) = 1;
    end
end


% ---------- 8. Check if a PAZ Alert extension is necessary: ----------
% If an alert is issued on a particular target, the duration of that
% alert is extended to at least last the duration of the aural
% annunciation, or the 'config.reAlertDelay' value.

if ~isempty(target.t_PAZ_Alerts_On)
    if target.PAZ_Alert_Status(end) == 0 && ...
       target.tracker_t(end) - target.t_PAZ_Alerts_On(end) <= ...
       config.reAlertDelay

        target.PAZ_Alert_Status(end) = 1;
    end
end
```

```
% ---------- 9. Check if a CAZ Alert extension is necessary: ----------
% If an alert is issued on a particular target, the duration of that
% alert is extended to at least last the duration of the aural
% annunciation, or the 'config.reAlertDelay' value.

if ~isempty(target.t_CAZ_Alerts_On)
    if target.CAZ_Alert_Status(end) == 0 && ...
       target.tracker_t(end) - target.t_CAZ_Alerts_On(end) <= ...
       config.reAlertDelay

        target.CAZ_Alert_Status(end) = 1;
    end
end


% -------- 10. Check to see if a PAZ alert is being pronounced: --------
% If we've issued a PAZ alert in the past, are currently predicting
% a CAZ alert and the time since we've issued the PAZ alert is less
% than the re-Alert delay, the CAZ alert is turned off.

% If we've issued a PAZ alert in the past, ...
if ~isempty(target.t_PAZ_Alerts_On)
    % ... check to see whether we are trying to issue a PAZ and a CAZ
    % alert during this call of TSAA:
    if target.CAZ_Alert_Status(end) == 1 && ...
       target.PAZ_Alert_Status(end) == 1 && ...
       target.PAZ_Alert_Status(end-1) == 0

        target.CAZ_Alert_Status(end) = 0;

    % ... else, check to see whether we've issued a PAZ alert within
    % the reAlertDelay period:
    elseif target.CAZ_Alert_Status(end) == 1 && ...
       target.tracker_t(end) - target.t_PAZ_Alerts_On(end) <= ...
       config.reAlertDelay

        target.CAZ_Alert_Status(end) = 0;
    end

% If, on the other hand, we've never issued a PAZ alert, we wouldn't
% issue a CAZ alert yet either:
elseif isempty(target.t_PAZ_Alerts_On)
    target.CAZ_Alert_Status(end) = 0;
end


% As a last step, if we issued an alert during this TSAA call, save the
% current time for future use:
if length(target.PAZ_Alert_Status) > 1 && ...
   target.PAZ_Alert_Status(end) == 1 && ...
   target.PAZ_Alert_Status(end-1) == 0

    target.t_PAZ_Alerts_On = [target.t_PAZ_Alerts_On ...
                              target.tracker_t(end)];
end
```

```
    if length(target.CAZ_Alert_Status) > 1 && ...
       target.CAZ_Alert_Status(end) == 1 && ...
       target.CAZ_Alert_Status(end-1) == 0

        target.t_CAZ_Alerts_On = [target.t_CAZ_Alerts_On ...
                                  target.tracker_t(end)];
    end

    % Similarly, if we turned an alert Off, store for later use:
    if length(target.PAZ_Alert_Status) > 1 && ...
       target.PAZ_Alert_Status(end) == 0 && ...
       target.PAZ_Alert_Status(end-1) == 1

        target.t_PAZ_Alerts_Off = [target.t_PAZ_Alerts_Off ...
                                  target.tracker_t(end)];
    end

    if length(target.CAZ_Alert_Status) > 1 && ...
       target.CAZ_Alert_Status(end) == 0 && ...
       target.CAZ_Alert_Status(end-1) == 1

        target.t_CAZ_Alerts_Off = [target.t_CAZ_Alerts_Off ...
                                  target.tracker_t(end)];
    end

end
```

Published with MATLAB® R2013b

Contents

TSAA Aircraft Class

```
classdef tsaa_aircraft_class
```

Introduction

This class is used by the TSAA.m script to store the information necessary for TSAA to perform its function. Instances of it will be created for the ownship as well as for each active target. It should be noted that this class is written as to maintain all history of a given target and does not actively manage the amount of data it stores. In a real-life implementation, additional consireations to prevent memory over-runs are required.

Class Properties

A list of the states that are stored and used by TSAA. It contians variables for the states that are received from a target, a variable that stores the aircraft's currently predicted future trajectory (over-written every time a conflict search is performed), as well as a list of variables that store the results from the conflict search once it is performed. The units are meters, seconds, meters per second, radians and radians per second:

```
    properties
        % Properties that store aircraft states:
        num;              % ID of the aircraft
        tracker_t;        % Time stamp of the aircraft's last (extrapolated)
                          % position
        report_t;         % Time stamp of the last time a report was received
                          % this aircraft (prior to extrapolation)
        x;                % X position (meters)
        y;                % Y position (meters)
        z;                % Altitude (in meters)
        xdot;             % East Velocity (meters/sec)
        ydot;             % North Velocity (meters/sec)
        zdot;             % Vertical rate (meters/sec)
        psi;              % Heading (radians)
        psidot;           % Turn rate (radians/sec)

        % The property that stores the currently projected trajectory:
        traj;

        % Properties that identify the alert state of the target (only
```

```
        % populated for targets everytime TSAA is called in 'detect' mode):
        t_to_PAZ_conflict;   % Time into the future at which the first PAZ
                             % conflict is predicted
        t_to_CAZ_conflict;   % Time into the future at which the first CAZ
                             % conflict is predicted
        PAZ_Alert_Status = 0;% Indicates whether this target is in (PAZ)
                             % alert state. Initialized as Alert State Off.
        CAZ_Alert_Status = 0;% Indicates whether this target is in (CAZ)
                             % alert state. Initialized as Alert State Off.
        t_PAZ_Alerts_On;     % Stores the time at which PAZ alerts are
                             % issued
        t_CAZ_Alerts_On;     % Stores the time at which CAZ alerts are
                             % issued
        t_PAZ_Alerts_Off;    % Stores the time at which PAZ alerts are
                             % turned off
        t_CAZ_Alerts_Off;    % Stores the time at which CAZ alerts are
                             % turned off
    end
```

Class Methods

A list of the methods that perform operations on the properties stored above to generate the data necessary for TSAA:

```
    methods
```

Class Constructor

Generates an instance of the tsaa_aircraft_class. If the constructor is called with 'Ownship' as input data, the object is generated but no data is stored in it's properties. If it is called with state information, the object is constructed and the state data is stored in the properties in one call.

```
        function obj = tsaa_aircraft_class(varargin)
            if ~strcmp(varargin, 'Ownship')
                obj.num = varargin{1}.num;
                obj = update(obj, varargin{1});
            elseif strcmp(varargin, 'Ownship')
                obj.num = 1;
            end
        end
```

Update Function

This function updates the object when new state data becomes available:

```
        function obj = update(obj, state_data)
           obj.tracker_t = [obj.tracker_t state_data.t];
           obj.x = [obj.x state_data.x];
           obj.y = [obj.y state_data.y];
           obj.z = [obj.z state_data.z];
           obj.xdot = [obj.xdot state_data.xdot];
           obj.ydot = [obj.ydot state_data.ydot];
           obj.psi = [obj.psi atan2(obj.ydot(end), obj.xdot(end))];
```

```
        % Since the ownship is not tracked, the time stamps of its state
        % data will be the time stamps of the reports:
        if state_data.num == 1
            obj.report_t = [obj.report_t state_data.t];
        else
            obj.report_t = [obj.report_t state_data.tor];
        end
    end
```

Trajecotry Propagation

This function propagates the trajecotry for this TSAA call based on the data that is currently available in the objects properties. The trajecotry is propagated into the future for the number of seconds defined in the config file. The final trajecotry will be an array of the following 7 columns:

1. Time into the future in seconds
2. Predicted x position
3. Predicted y position
4. Predicted altitude
5. Predicted x velocity
6. Predicted y velocity
7. Predicted z velocity

```
    function obj = propagate_trajectory(obj, config)
        % Propagates a trajecotry that is 'lookahead' seconds long with
        % points calculated every 'dt' seconds.

        % First, necessary states are calculated (Note: this is only
        % necessary if the turn and vertical rate are not provided via
        % the received state data.)
        obj.zdot = [obj.zdot estimate_zdot(obj, config)];
        obj.psidot = [obj.psidot estimate_psidot(obj, config)];

        % For the ownship we allocate 7 columns and for targets 19:
        if obj.num(end) == 1
            trajectory = zeros(length(0:config.dt:config.lookahead), 7);
        else
            trajectory = zeros(length(0:config.dt:config.lookahead),19);
        end

        spd = sqrt(obj.xdot(end)^2 + obj.ydot(end)^2);
        trajectory(1, 1:7) = [0 ...
                        obj.x(end) ...
                        obj.y(end) ...
                        obj.z(end) ...
                        obj.xdot(end) ...
                        obj.ydot(end) ...
                        obj.zdot(end)];
        idx = 2;
        for t = config.dt:config.dt:config.lookahead
            trajectory(idx, 1:7) = [t ...
                 trajectory(idx-1,2) + trajectory(idx-1,5)*config.dt ...
```

```
                trajectory(idx-1,3) + trajectory(idx-1,6)*config.dt ...
                trajectory(idx-1,4) + obj.zdot(end)*config.dt ...
                spd*cos(obj.psi(end) + t*obj.psidot(end)) ...
                spd*sin(obj.psi(end) + t*obj.psidot(end)) ...
                obj.zdot(end)];
            idx = idx + 1;
        end

        % Store the trajectory that was just created in the current
        % object:
        obj.traj = trajectory;
    end
```

Vertical Rate Estimation

A basic vertical rate estimator to calculate the vertical rate needed to project the future trajecotry. This is only needed if no vertical rate is provided as an input to TSAA:

```
    function zdot = estimate_zdot(obj, config)
        % Find the number of data points that are used to calculate
        % zdot. It's at least the number of reports currently
        % available or at most the number of data points defined in
        % config. The -1 is necessary becuase we're also counting the
        % current update:
        points = min(config.numberOfUpdatesVert, length(obj.z))-1;

        % We need a minimum of three altitudes to estimate a vertical
        % rate using a linear fit approach:
        if points >= 3
            time_stamps = obj.tracker_t(end-points:end) - ...
                          obj.tracker_t(end);
            altitudes = obj.z(end-points:end);
            xmean = mean(time_stamps);
            ymean = mean(altitudes);
            x_values = time_stamps - xmean;
            y_values = altitudes - ymean;
            zdot = dot(x_values, y_values)/dot(x_values, x_values);

        % If not enought data is available for a linear fit, the
        % instantaneous climb rate is used.
        elseif points == 2
            zdot = (obj.z(end) - obj.z(end-1))/(obj.tracker_t(end) - ...
                   obj.tracker_t(end-1));
        else
            zdot = 0;
        end
    end
```

Turn Rate Estimation

A basic turn rate estimator to calculate the turn rate needed to project the future trajecotry. This is only needed if no turn rate is provided as an input to TSAA:

```
    function psidot = estimate_psidot(obj, config)
```

```
            % Find the number of data points that are used to calculate
            % psidot. It's at least the number of reports currently
            % available or at most the number of data points defined in
            % config. The -1 is necessary becuase we're also counting the
            % current update:
            points = min(config.numberOfUpdatesTurn, length(obj.psi))-1;

            % We need a minimum of two headings to estimate a turn rate:
            if length(obj.psi) >= 2
                delta_psi = diff(obj.psi(end-points:end));
                delta_t = diff(obj.tracker_t(end-points:end));

                % Check to see that all delta_psi values are between -pi and
                % +pi:
                idx = delta_psi < -pi;
                delta_psi(idx) = delta_psi(idx) + 2*pi;

                idx = delta_psi > pi;
                delta_psi(idx) = delta_psi(idx) - 2*pi;

                psidot = mean(delta_psi./delta_t);

            else
                psidot = 0;
            end

        end
```

```
    end
```

```
end
```

Published with MATLAB® R2013b