https://ntrs.nasa.gov/search.jsp?R=20190027045 2019-09-26T19:02:18+00:00Z

Provisional chapter

A Fuzzy Logic Approach for Separation Assurance and Collision Avoidance for Unmanned Aerial Systems

Brandon Cook, Tim Arnett and Kelly Cohen

Additional information is available at the end of the chapter

Abstract

In the coming years, operations in low altitude airspace will vastly increase as the capabilities and applications of small unmanned aerial systems (sUAS) continue to multiply. Therefore, finding solutions to managing sUAS in highly congested airspace will facilitate sUAS operations. In this study, a fuzzy logic-based approach was used to help mitigate the risk of collisions between aircraft using separation assurance and collision avoidance techniques. The system was evaluated for its effectiveness at mitigating the risk of mid-air collisions between aircraft. This system utilizes only current state information and can resolve potential conflicts without knowledge of intruder intent. The avoidance logic was verified using formal methods and shown to select the correct action in all instances. Additionally, the fuzzy logic controllers were shown to always turn the vehicles in the correct direction. Numerical testing demonstrated that the avoidance system was able to prevent a mid-air collision between two sUAS in all tested cases. Simulations were also performed in a three-dimensional environment with a heterogeneous fleet of sUAS performing a variety of realistic missions. Simulations showed that the system was 99.98% effective at preventing mid-air collisions when separation assurance was disabled (unmitigated case) and 100% effective when enabled (mitigated case).

Keywords: fuzzy logic, UAS, collision avoidance, separation assurance, formal methods, satisfiability modulo theories

1. Introduction

In recent times, there have been substantial advances in the capability of mobile robots in several aerospace applications. These advances include autonomous intelligence, surveillance, and reconnaissance (ISR) efforts [1], aerial firefighting [2], and aerial delivery services [3]. However, despite the potential benefits, these advancements are currently being under-utilized due to

INTECH
open science | open minds

© 2017 The Author(s). Licensee InTech. This chapter is distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

several unresolved safety issues with integrating these platforms into the National Airspace System (NAS). As a result of these shortcomings, there is a need to develop algorithms that allow a heterogeneous team of small unmanned aerial systems (sUAS) to interact autonomously and perform time-critical tasks in complex environments. As the applications and capabilities of sUAS continue to proliferate, it is imperative to address the safe integration of these vehicles into the NAS.

Most of the work in this area deals with separation assurance, as it typically takes priority in NAS conflict resolution scenarios [4, 5]. However, most methods necessitate the communication of state information between the vehicles in order to properly select resolution actions [6]. For collision avoidance, several intelligent systems have been developed with promising results [7–12], but few have also shown behavioral verification using formal methods [13, 14].

To facilitate real-time control of a large number of sUAS, a fuzzy logic approach was implemented. This approach was utilized to mitigate the risk of losses of separation and ultimately collisions, between the sUAS. In order to generate scenarios to test the sUAS's ability to avoid collisions, a realistic simulation environment was created. This simulation environment was developed in a modular fashion, such that various algorithms could be implemented to coordinate sUAS maneuvers. This enables various vehicle platforms, sensor models, software packages, and traffic management methodologies to be tested and evaluated.

The main goal of this research is to develop a high-level concept of operations for a UAS traffic management (UTM) system. This system must address the challenges of collision avoidance and separation assurance. Each of these platforms will utilize fuzzy logic controllers (FLCs) to enable real-time decision-making and dynamic control. Additionally, the confidence in correct decision-making and avoidance control outputs needs to be extremely high. Therefore, formal methods were employed for behavioral verification.

The remainder of the paper is as follows. In Section 2, background material on some of the methods and tools used in this work is described. Section 3 details the proposed solution, which includes the separation assurance and collision avoidance methods. This includes detailed development procedures for the decision-making and fuzzy avoidance controllers. Section 4 presents the methodology for implementing and evaluating the decision-making and fuzzy avoidance controllers using formal methods. Section 5 then explains the test cases, their implementations, and an overview of the simulation environment and constraints. Section 6 presents results from the formal methods evaluations and simulation runs, and finally, Section 7 discusses conclusions and opportunities for future work.

2. Background

2.1. Hybrid fuzzy systems

Most fuzzy inference systems (FISs) involve multiple operations that associate inputs to outputs based on multiple if-then rules that are resolved to a singleton. The output space is typically nonlinear and difficult to describe as a function of the input variables. However, by constraining the FIS to have particular properties and association methods, an explicit expression can be more easily found.

Hybrid systems are systems that have regions of continuous behavior separated by discrete transitions [15]. This is analogous to a subset of FISs that contain membership functions that are constrained to a finite domain. Such FISs can be represented as hybrid systems after an explicit expression is found. This expression maps an input set to an output set using a set of mathematical functions. This is useful due to the number of low-level tools that have been developed for analyzing hybrid systems. Among these are formal methods tools [16], which are described in the following section.

2.2. Formal methods

In systems such as UTM collision avoidance algorithms, the level of confidence that they will always behave as intended needs to be extremely high. Typical methods for evaluating these algorithms usually involve simulation, but simulation and other numerical methods can miss critical cases that result in undesired behavior. To increase the confidence that the avoidance algorithms presented work as intended, formal methods were employed. Formal methods are defined by NASA as "mathematically rigorous techniques and tools for the specification, design, and verification of software and hardware systems." [17] There are numerous types of tools that fall under this definition, but in this work, satisfiability modulo theories (SMT) solvers and model checkers were used.

SMT solvers are tools that extend the Boolean Satisfiability (SAT) problem to first order logic (FOL) sentences and incorporate other theories for evaluating the truth assignments of variables (real values, bitvectors, etc.). If a behavior can be described in FOL, it can be encoded and evaluated by an SMT solver to find truth assignments that violate this behavior. If a behavior is found, it is returned as satisfying the behavioral specification. If there are no possible assignments to the variables that render the specification true, it is then said to be unsatisfiable. Therefore, "safety" properties, properties which should always hold true, can be evaluated by negating a specification that encapsulates the respective behavior. If a satisfying case is found for the negated specification, this means that there are conditions that violate the original specification. If no satisfying cases are found, then the original specification will hold for all possible conditions.

Model checkers are tools that exhaustively check the states of a system to search for combinations of variable assignments that violate behavioral specifications. In finite state systems, they use deductive proofs, and in infinite state systems, they can use inductive methods. These tools can also use SAT or SMT solvers in conjunction with their own search methods for finding counterexample cases. Encoding safety properties in model checkers is slightly different, however, as model checkers typically use some type of temporal operator in conjunction with logical sentences. However, there are methods for relating quantified FOL sentences for safety properties to temporal representations for use in model checkers [18].

In this work, an infinite-state model checker named JKind [19, 20] was used. JKind is a Java implementation of the Kind model checker which uses *k*-induction. To evaluate the truth assignments for variables within each state, JKind employs SMT solvers. The SMT solver used for this work is Z3 [21], a state-of-the-art SMT solver developed by Microsoft.

3. Proposed solution

To ensure that two or more sUAS do not collide with one another, an intruder avoidance system was developed. This avoidance problem is broken down into two sub-systems: strategic (separation assurance) and tactical (collision avoidance). The strategic separation assurance platform uses a centralized approach to coordinate trajectory modifications for sUAS to ensure that vehicles do not get too close to one another. This separation assurance technique is employed when two or more vehicles come within 0.4 nmi laterally of one another (separation alert threshold), 100 ft. vertically, and are predicted to have a loss of separation (LOS), defined by when vehicles come within 0.1 nmi laterally and 100 ft. vertically of one another. This LOS threshold was determined based on the characteristics of the vehicle platforms and feasible sensing abilities of sUAS. Based on the system constraints, the avoidance platform would have roughly 2 sec to resolve maximum closure rate encounters. If the separation assurance system fails to prevent an LOS, the vehicles will employ their onboard sense and avoid systems to prevent a mid-air collision. A mid-air collision occurs when two vehicles come within 60 m of one another. This collision threshold is intentionally conservative to introduce a notion of spatial uncertainty. Since the sensor models provide perfect state information, as described in Section 5, all vehicle locations are precisely known. For this study, the collision avoidance platform uses a de-centralized approach, that is, all vehicles attempt to avoid intruding vehicles independently. Thus, no communication between aircraft is available (i.e., uncoordinated maneuvers). In **Table 1**, the various distance thresholds used to describe the separation boundaries are shown.

Prior to presenting the details of each avoidance sub-system, Section 3.1 provides an overview of the avoidance system architecture. This overarching logic is used to determine when a vehicle should perform an avoidance maneuver. When deemed necessary, the system will activate the appropriate avoidance platform. In each avoidance platform, a set of heuristics are used to determine the appropriate action to resolve a conflict. These details are presented in Sections 3.2 and 3.3. Once the appropriate action has been decided, an FLC is used to control the vehicle's turn rate in the desired direction. The details of each FLC are shown in Section 3.4.

Finally, it is important to note that these two sub-systems use different approaches when trying to resolve conflicts between aircraft. This is primarily due to the overall purpose each sub-

Threshold label	Lateral distance	Vertical distance
Separation alert	0.4 nmi	100 ft.
LOS	0.1 nmi	100 ft.
Collision	60 m	50 ft.

Table 1. Separation threshold values.

system serves, and the information that is available to each. If vehicles are reporting their state information to a ground-based station, a centralized separation assurance platform could be used to coordinate a trajectory modification to one or more of the vehicles. Thus, coordinated maneuvers are possible. However, when two vehicles are within seconds away from a collision, minimizing the time between sensing the vehicle and performing an action is critical. Therefore, when the collision avoidance system is activated, the vehicles must independently choose the appropriate action using onboard processors. In these collision avoidance scenarios, there is no communication between aircraft. Thus, each sub-system requires a different set of rules to determine the appropriate action.

3.1. Overarching control logic

The overarching control logic determines whether to perform a separation assurance maneuver, activate the collision avoidance system, or allow vehicles to continue along their desired trajectories. This logic is shown in flow chart form in **Figure 1**. First, the system will find the distance separating all aircraft pairs. With this information, a calculation is made to see how much time can pass prior to two vehicles coming within 0.4 nmi. To calculate this value, the current separation, minus the 0.4 nmi threshold, is divided by the maximum closure rate of the aircraft pair. Therefore, if both vehicles moved directly toward one another at their maximum allowable speeds, this is the time it would take them to reach the 0.4 nmi separation threshold. This future time is known as the "time threshold", as shown in **Figure 1**. Using this time threshold, if two vehicles cannot possibly be within 0.4 nmi of one another, the system will not unnecessarily check if the two aircraft are in conflict. Rather, it remains idle between checks to improve the performance of the system.

Once enough time has passed and an aircraft pair reaches their assigned time threshold, the system will again check their separation. If the two aircraft are still more than 0.4 nmi apart, a new time threshold is calculated and set. However, if the aircraft pair has reached the 0.4 nmi threshold, it will next check to see if an LOS has occurred. If the vehicles have violated the 0.1 nmi LOS threshold, the collision avoidance system is enabled. Otherwise, the separation assurance system may be needed to ensure that two vehicles do not have an LOS. This decision is based on two criteria: if the predicted closest point of approach (pCPA) creates an LOS and if the time to LOS (tLOS) is within 2 min.

If both criteria are met, a final check is used to see if the aircraft pair has already been assigned a separation assurance maneuver. If neither vehicle has been assigned a maneuver to avoid an impending LOS, a resolution advisory is sent from the centralized system to one of the sUAS. However, if the sUAS was already assigned a maneuver and is currently in the middle of its resolution, a check is used to see if turning back toward its preferred heading will cause another predicted LOS. If resuming its originally intended mission will not cause an LOS, it will do so, otherwise, the sUAS will continue on its current bearing.

If neither criterion (pCPA and tLOS) is met, then no separation assurance command is given and the aircraft will continue toward its respective target using its navigation controller. After determining the appropriate separation assurance action, or deciding that no action is needed, the algorithm calculates another time threshold for each aircraft pair.

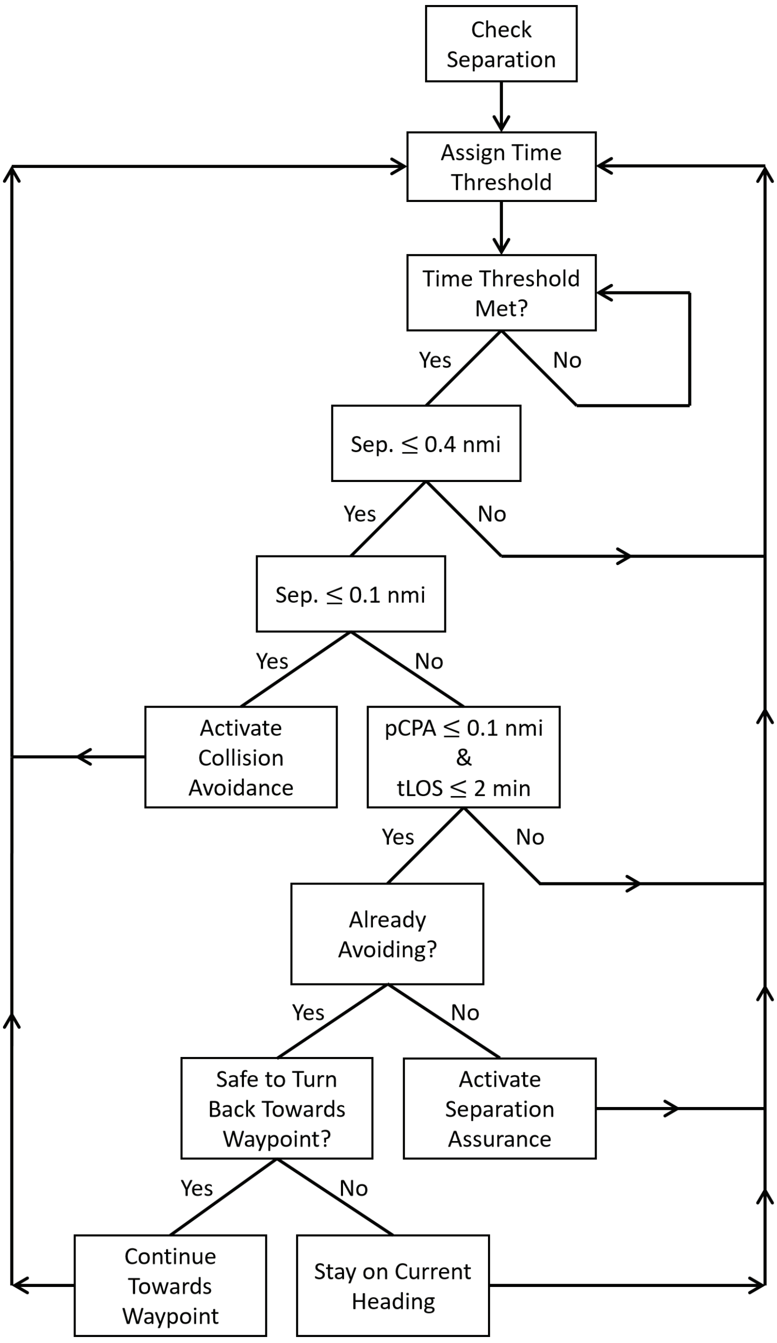

Figure 1. Control logic flow chart.

3.2. Separation assurance logic

The centralized separation assurance system observes only the current position, heading, and velocity of each vehicle. With this limited information, vehicle intent is unknown. Therefore, the system must be robust to dynamic scenarios and resolve conflicts without the knowledge of other vehicles' goals.

The separation assurance platform will be enabled if three criteria are met: separation less than 0.4 nmi, the pCPA is less than 0.1 nmi, and the tLOS is within 2 min. If two aircraft meet all three of these criteria, the separation assurance platform will activate and assign one of the vehicles a new trajectory in an effort to avoid the predicted LOS.

To determine what action the separation assurance platform should use to avoid a potential LOS, a series of conflict classification techniques are used. For this study, three pieces of information are used to classify all conflict scenarios: relative heading, relative angle, and crossing time. These parameters can be described using the more common parameters: location, speed, and heading. In **Figure 2**, a sample conflict scenario is shown. Here, the triangular objects each represent an sUAS, the arrows represent the velocities of each aircraft (both magnitude and direction), and the "x" represents the heading intersection point location. The heading intersection point is not to be confused with the pCPA. It is simply the point where the projected headings intersect with one another. For this example, the vehicle with the small circle represents aircraft 1, and the other represents aircraft 2.

The relationship used to describe the heading of vehicle 2 relative to vehicle 1's perspective is shown in Eq. (1).

$$R_{H_1} = H_2 - H_1 \tag{1}$$

where H_1 is the heading of vehicle 1, H_2 is the heading of vehicle 2, and R_{H_1} is the relative heading from vehicle 1's perspective. In this study, $0° \leq R_{h_i} < 360°$ for all vehicles, where i represents the index for each vehicle. Therefore, if $H_1 > H_2$, a 360° phase shift must be added to R_{H_1} to remain within the constrained range. Computing the relative heading for the example shown in **Figure 2**, R_{H_1} would be 140° (i.e., moving to the left with respect to vehicle 1), whereas, from vehicle 2's perspective, R_{H_2} would be roughly 220° after applying the 360° phase shift (i.e., moving to the right with respect to vehicle 2).

Similarly, the relative angle between two vehicles describes the relative position of one vehicle with respect to the other. This relationship has been shown in Eq. (2).

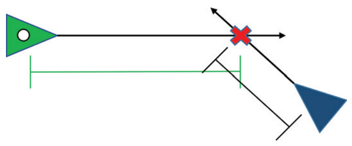

Figure 2. Conflict scenario classification information.

$$R_{A_1} = \tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right) - H_1 \tag{2}$$

where x_2 and y_2 are the two-dimensional coordinates of vehicle 2 in the global frame, x_1 and y_1 are the two-dimensional coordinates of vehicle 1 in the global frame, H_1 is the heading of vehicle 1, and R_{A_1} is the relative angle of vehicle 2 from vehicle 1's perspective. Like the relative heading, the relative angle is constrained. Here, we constrain the relative angle by the following relationship: $-180° \leq R_{A_1} \leq 180°$. Therefore, if the relative angle is less than $-180°$, a 360° phase shift is added to meet this constraint, or if the angle is greater than 180°, a 360° phase shift is subtracted. Again, using **Figure 2** as an example, R_{A_1} would be roughly $-15°$ after subtracting a 360° phase shift (i.e., vehicle 2 is to the right of vehicle 1 from vehicle 1's perspective), and R_{A_2} would be roughly 25° (i.e., vehicle 1 is on the left of vehicle 2 from vehicle 2's perspective).

Lastly, the crossing time, t_1, can be defined by the relationship shown in Eq. (3). This relationship defines how long it would take for vehicle 1 to reach the heading intersection point, as shown in **Figure 2**, if they remained on their current trajectory.

$$t_1 = sign\left(\vec{V_1} \cdot \vec{C_1}\right)\frac{c_1}{v_1} \tag{3}$$

where *sign* is a function used to determine the sign (positive or negative) of an expression, $\vec{V_1}$ is the velocity of vehicle 1, $\vec{C_1}$ is the vector used to describe the location of the heading intersection point relative to the vehicle's current position, v_1 is the speed of vehicle 1, and c_1 is the magnitude of the vector $\vec{C_1}$.

With these relationships, all possible encounter scenarios can be described. To aid in understanding what each of these parameters represent, **Table 2** describes, in linguistic terms, what each range of values represents in the physical conflict scenarios. Here, the term crossing point

Parameter	Range	Meaning
Relative heading	$0° < R_{H_1} < 180°$	Moving from right to left
	$180° < R_{H_1} < 360°$	Moving from left to right
	$R_{H_1} = 0°$	Same direction
	$R_{H_1} = 180°$	Head-on
Relative angle	$R_{A_1} > 0°$	On left
	$R_{A_1} < 0°$	On right
	$R_{A_1} = 0°$	Straight ahead
Time to crossing point	$t_1 > 0$	Crossing point in front
	$t_1 < 0$	Crossing point behind
	$t_1 > t_2$	Farther from crossing point
	$t_1 < t_2$	Closer to crossing point

Table 2. Linguistic descriptions of encounter scenarios.

is synonymous to the heading intersection point. Each of these descriptions has been listed to describe the motion, location, or crossing time of vehicle 2 from vehicle 1's perspective.

Although the primary goal of this system is to ensure safe separation of vehicles, it is also important to try and limit the number of unnecessary flight adjustments. This is particularly important when operating sUAS due to their typical limitations in power and endurance. Because vehicle intent is unknown in this study, a predicted LOS does not guarantee an LOS is imminent. Therefore, there is a tradeoff between using strict and relaxed criteria when determining if a trajectory modification is necessary. The criteria should be relaxed to ensure sUAS do not repetitively perform unnecessary adjustments but strict enough to ensure safe operation.

Aside from optimizing this time to predicted LOS threshold, a second way to limit the number of vehicles that divert from their desired flight paths is to assign vehicles priority. This priority assignment ranks all vehicles in conflict from highest priority to lowest priority. Therefore, the vehicle with the highest priority will continue on its preferred trajectory without modification. However, all vehicles with a lower priority must avoid all other vehicles with a higher priority.

To determine which aircraft has a lower priority, a series of evaluations are made. First, the system will be checked to see if the two aircraft are moving in a similar direction. If two vehicles have a heading within 5° of one another, that is, $355° \leq R_{H_1} < 360°$ or $0° \leq R_{H_1} \leq 5°$, the trailing aircraft will have lower priority. This encounter scenario can be seen in **Figure 3**. If several aircraft have similar headings, the aircraft furthest behind will be assigned the lowest priority so must avoid all other aircraft. However, the vehicle in the front of the group will have the highest priority and will disregard all other aircraft.

If the vehicles in conflict do not have similar headings (i.e., more than a 5° difference), the vehicle closest to its next waypoint is given priority. Since the separation assurance system logic does not use the location of a vehicle's next waypoint (i.e., intent is unknown), this priority assignment was simply a means to an end. In practice, the priority of each vehicle in these scenarios would be randomly assigned.

To predict whether an LOS will occur, the separation assurance algorithm uses the current location and velocity of each aircraft to calculate a projected flight path for each. Using these projected trajectories, the pCPA between the aircraft is found. If the pCPA will result in an LOS within the next 2 min, a resolution is calculated and employed to prevent the predicted LOS.

Recalling the sample encounter scenario shown in **Figure 2**, the definitions described by Eqs. (1) through (3), and the constraints shown in **Table 2**, all encounter scenarios can be described. **Figure 4** depicts all the possible conflict scenarios when vehicle 2 is located to the right of vehicle

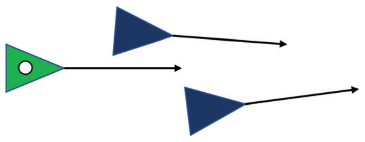

Figure 3. Encounter example with relative heading within 5°.

1. In each diagram, the aircraft with the small circle represents vehicle 1 and the other is vehicle 2. Thus, all parameters used to describe a particular conflict scenario are from the perspective of vehicle 1. Within the scope of the separation assurance system, vehicle 2 is classified as the vehicle that has been assigned the higher priority. Thus, vehicle 1 (lower priority) must perform a maneuver to prevent an LOS.

In **Figure 4(a)**, vehicle 2 is moving to the left, vehicle 1 is approaching from behind, and vehicle 1 is more than 45 sec closer to the heading intersection point. In this case, vehicle 1 would decide to go in front of vehicle 2. However, if vehicle 1 is not at least 45 sec closer, it will go behind.

In **Figure 4(b)**, vehicle 2 is still going to the left, but in this case, it is coming toward vehicle 1. In these scenarios, vehicle 1 must be more than 30 sec closer to the intersection point to go in front. In **Figure 4(c)**, vehicle 2 is to the right of vehicle 1 but is also going to the right. In these instances, the logic will determine vehicle 1 should turn left to avoid a potential LOS. This also holds for when vehicle 2 is located directly in front of vehicle 1. If, however, vehicle 2 is located on the left of vehicle 1 and going left, it will be instructed to turn right. (NOTE: The 30 and 45 sec buffers were selected after testing a handful of design iterations. Optimizing these buffer thresholds is left to future work.)

In **Figure 4(d)**, vehicle 2's heading is parallel and coming toward vehicle 1. If vehicle 2 is directly in front of, or to the left of, vehicle 1, the logic will instruct vehicle 1 to turn right. However, if vehicle 2 is located to the right of vehicle 1, it will be instructed to turn left.

To prevent an aircraft from prematurely exiting an avoidance maneuver, the system checks if reverting to the navigation controller generates another predicted LOS. Since the avoidance controller has only local sensor knowledge (i.e., $-90^\circ \leq R_{A_1} \leq 90^\circ$), switching back to the navigation controller can result in a turning action that generates another predicted LOS. If turning back to its desired target would create another predicted LOS, the avoiding aircraft will continue with its trajectory until authorized to resume its desired mission.

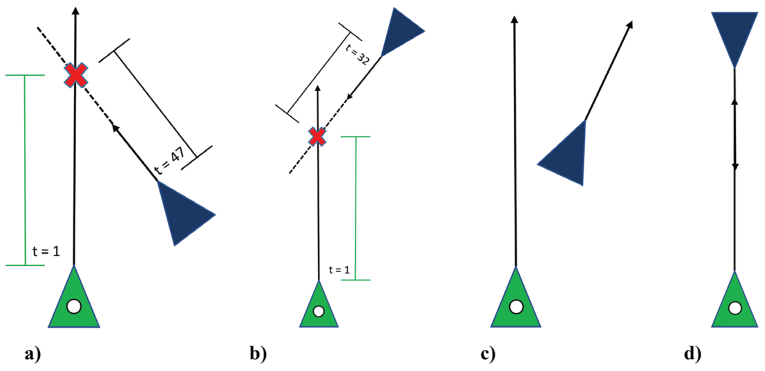

Figure 4. Separation assurance conflict scenario classifications: (a) from behind, (b) coming toward, (c) diverging, and (d) head-on.

Condition	Relative angle or time to cross	Connector	Relative heading	Action
IF	Any angle within sensor range ($-90^\circ \le R_{A_1} \le 90^\circ$)	AND	Similar direction ($R_{H_1} < 5^\circ$ OR $R_{H_1} > 355^\circ$)	Go behind
ElseIF	On right OR straight ahead ($R_{A_1} \le 0^\circ$)	AND	Going right ($180^\circ < R_{H_1} < 360^\circ$)	Go behind
ElseIF	On left ($R_{A_1} > 0^\circ$)	AND	Going left ($0^\circ < R_{H_1} < 180^\circ$)	Go behind
ElseIF	On right ($R_{A_1} < 0^\circ$)	AND	Head-on ($R_{H_1} = 180^\circ$)	Turn left
ElseIF	On left OR straight ahead ($R_{A_1} \ge 0^\circ$)	AND	Head-on ($R_{H_1} = 180^\circ$)	Turn right
ElseIF	I'm more than 45+ seconds closer ($t_1 < (t_2 - 45)$)	AND	Approaching from behind ($R_{H_1} < 90^\circ$ OR $R_{H_1} > 270^\circ$)	Go in front
ElseIF	I'm NOT 45+ seconds closer ($t_1 \ge (t_2 - 45)$)	AND	Approaching from behind ($R_{H_1} < 90^\circ$ OR $R_{H_1} > 270^\circ$)	Go behind
ElseIF	I'm more than 30+ seconds closer ($t_1 < (t_2 - 30)$)	AND	Coming towards ($90^\circ \le R_{H_1} \le 270^\circ$)	Go in front
ElseIF	I'm NOT 30+ seconds closer ($t_1 \ge (t_2 - 30)$)	AND	Coming towards ($90^\circ \le R_{H_1} \le 270^\circ$)	Go behind

Table 3. Summary of separation assurance logic.

A summary of the separation assurance logic can be found in **Table 3**. For all cases, the crossing time is strictly positive. That is, the absolute value of the true crossing time found using Eq. (3) is used for all separation assurance logic.

3.3. Collision avoidance logic

When two vehicles have an LOS, are converging, and are within one another's sensor ranges, the collision avoidance system will be activated. In this study, each sUAS will attempt to avoid all intruders within its sensor range; therefore, no vehicle priority will be assigned. Like the approach used for the separation assurance platform to classify conflict scenarios, the collision avoidance system will use the same inputs to decide the appropriate action (i.e., relative angle, relative heading, and time to crossing point). Here, it is important to note that no two vehicles can communicate with one another. Therefore, the same logic is used onboard each system independently. This means that from each vehicle's perspective, we need to ensure that both vehicles will choose complementary actions, that is, the action will not force the vehicles to turn toward one another.

In **Figure 5**, all possible encounter scenarios are shown. Here, the black triangle and arrow represent the "ownship" (vehicle 1) location and heading respectively, the filled circle represents the "intruder" (vehicle 2) location, the dashed line connecting the two vehicles represents the relative position, and the other two dashed lines represent intruder headings that are either parallel or perpendicular to the ownship's heading. In this figure, the intruder can have any heading between 0° and 360°.

Using these dashed lines to divide the possible intruder heading into cases, the geometry of each encounter scenario can be broken down into twelve cases, provided that two of the three dashed lines do not coincide with one another. In **Figure 5(a)**, the twelve cases are depicted: two cases where the intruder has a parallel relative heading (vertical line), two cases where the headings are perpendicular (horizontal line), two cases where the intruder heading is directly toward or away from the ownship position (line connecting the two vehicles), and all headings that lie in between these angles each count as one case (i.e., six angle ranges in between the lines). If two of the three dashed lines coincide, this possible geometric space reduces to eight possible cases, as shown in **Figure 5(b)** and **c**).

Now that the geometric configurations have been defined, let us now introduce the third characteristic, time to heading intersection point. Unlike the separation assurance platform, the time to the heading intersection point, or "crossing time," can be either positive or negative. Therefore, if the crossing point lies behind the vehicle, the crossing time becomes negative. Using

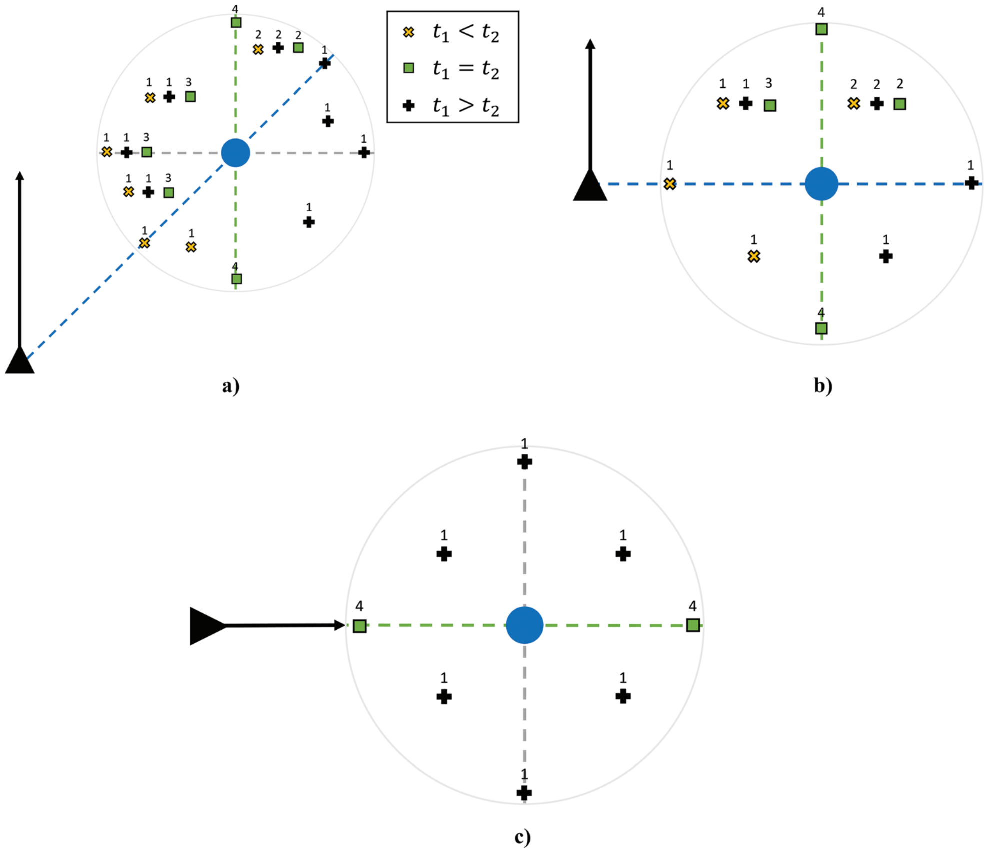

Figure 5. Classification of all possible encounter scenarios: (a) intruder not straight ahead or on side, (b) intruder on side, (c) intruder straight ahead.

the crossing time, there are up to three possible new situations for each of the twelve cases (or eight cases): the times are equal, the ownship crossing time is greater than that of the intruder, or the intruder crossing time is greater than that of the ownship. All possible crossing time scenarios based on the relative heading and angle have been shown. All instances where the ownship can have a crossing time less than the intruder crossing time have been marked by an "x." The black cross represents scenarios where the intruder crossing time can be less than the ownship crossing time. Finally, the square represents the situations where the two vehicles can have equal crossing times.

Using these three designations, all pairwise encounters where an intruder is not directly in front of or beside the ownship can be described by 20 possible cases, as shown in **Figure 5(a)**. In **Figure 5(b)**, cases where the intruder is directly beside the ownship are shown, resulting in 12 possible cases. Lastly, if the intruder is directly in front of the ownship, 8 additional cases can be attained, as shown in **Figure 5(c)**. Thus, a total of 40 cases can be attained using these three parameters to describe the pairwise encounter space.

Knowing that 40 possible cases have been shown, a total of four conflict classifications can be used to solve all possible encounter scenarios. In **Figure 5**, the number above each icon represents which of the four conflict classifications the scenario belongs. In **Figures 6** and **7**, the four conflict scenarios have been shown. In each figure, the ownship (vehicle 1) location is marked by a black triangle and its heading is designated by the black arrow. The small circle represents the location of the intruder (vehicle 2). This intruder can have any heading, but the

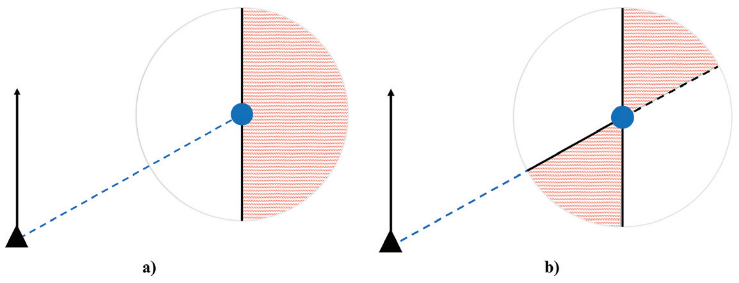

Figure 6. Conflict classification #1 where $(t_1 > 0) \vee (t_2 > 0)$ and: (a) $t_1 < t_2$, (b) $t_1 > t_2$.

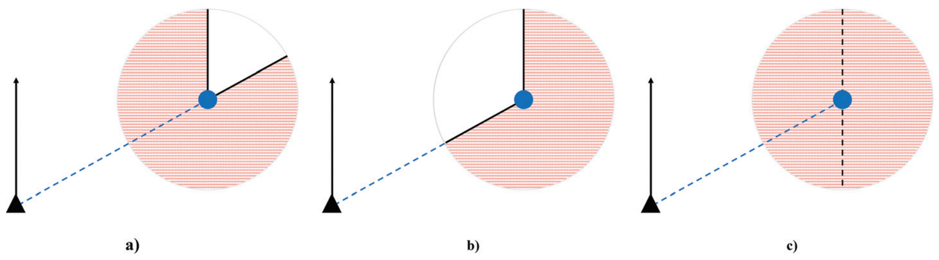

Figure 7. Conflict classifications #2, #3, and #4: (a) $(t_1 < 0) \wedge (t_2 < 0)$, (b) $(t_1 = t_2) \wedge (t_1 \neq \infty) \wedge (t_1 > 0)$, (c) $(t_1 = t_2) \wedge (t_1 = \infty)$.

different headings have been separated into different sections, as designated by the shaded regions. In each figure, the shaded regions represent the intruder headings that are excluded by that particular encounter scenario, whereas, the unshaded regions are the possible intruder headings allowed by that encounter scenario. In addition, like **Figure 5**, there is a dashed line connecting the two vehicles to represent the relative position. Lastly, if a dashed black line is on the boundary of the included and excluded regions, this represents an inclusive boundary (i.e., that heading is included in the possible headings allowed), whereas, the solid black line between the regions represents an exclusive boundary.

Figure 6 shows the first conflict classification scenario. Here, at least one of the vehicles must have a positive crossing time (t_i), and they are not equal to one another. As shown in **Figure 6(a)**, the ownship crossing time must be strictly less than that of the intruder. Thus, the intruder crossing time can be negative, or equal to zero (i.e., pointed directly at the ownship). If the intruder heading passes into the excluded region, either both values are negative, or the ownship value must be greater than the intruder value. For all the cases that satisfy this relationship, the ownship would determine to go in front of the intruder. Including the instances where the intruder is to the left of the ownship.

In **Figure 6(b)**, the ownship crossing time must be strictly greater than that of the intruder crossing time. As seen in the figure, the boundary along the relative position line is excluded when the intruder is pointing toward the ownship (i.e., the ownship crossing time is zero, but the intruder crossing time is positive), violating the relationship. However, when the intruder is pointed directly away from the ownship, the boundary is inclusive. In all possible instances, the ownship will determine to go behind the intruder. This includes the cases when the intruder is located to the left of the ownship or is directly in front of the ownship (i.e., $t_2 = 0$). For all cases represented by conflict classification #1, if one vehicle decides to go in front, the other will decide to go behind, given that they are both within one another's sensor field of view.

In the second conflict classification, as shown in **Figure 7(a)**, both the intruder and the ownship have crossing times less than zero. These negative crossing times, and the fact that the vehicles cannot have the same heading, result in the intruder never sensing the ownship. In these scenarios, if the intruder is on the left, the ownship will turn right. But, if the intruder is on the right, the ownship will turn left.

In **Figure 7(b)**, the scenarios where both the ownship and the intruder have the same time to the crossing point are shown. In this classification case, the intruder cannot have the same heading as the ownship, thus it must be crossing the ownship's path. In addition, this case is restricted to instances where both vehicles have strictly positive crossing times. For these instances, regardless of whether the vehicle is on the left, or on the right, both vehicles will decide to turn right. Since the crossing times are equivalent, and there is no coordination of intent with the other vehicle, both vehicles must choose the same action.

There is one remaining aircraft orientation in the encounter space. This occurs when an intruder is parallel to the ownship, either traveling in the same or opposite direction. This scenario has been shown in **Figure 7(c)**. As seen from this figure, only intruder headings that lie on the dashed line are included. If the intruder is to the right of the ownship, each vehicle will turn to the left (given that the intruder can see the ownship as well). However, if the intruder is directly in front of, or to the left of the ownship, both vehicles will be instructed to turn right.

Although the above classifications describe all possible encounters between moving aircraft, the quad-rotor vehicles can stop and hover. Therefore, a final classification must be described. If an intruder is stationary, the intruder crossing time is set to negative infinity. When this is the case, if the intruder is to the left or directly in front of the ownship, the controller will instruct the vehicle to turn right. However, if the intruder is to the right of the ownship, the logic will instruct the vehicle to turn left.

Table 4 describes the different encounter scenarios using both linguistic and mathematical descriptions, as well as the decided actions. Each of the above conflict classification numbers numerically matches the respective cases in this table. However, the encounter scenario where the intruder vehicle is stationary is referred to as case 0. Like **Table 3**, the linguistic descriptions in **Table 4** represent how the ownship perceives the intruder. Furthermore, all values with the subscript 1 represent the ownship, whereas, all values with the subscript 2 represent the intruder. (NOTE: the * designation indicates that the crossing time is negative infinity for that vehicle.)

3.4. Fuzzy inference systems

Using the methodology described in Sections 3.2 and 3.3, four possible actions can be used to avoid an intruding sUAS: go behind, go in front, turn right, and turn left. When the command

Case	Position	Direction	Time to cross	Crossing point location	Action
0	On left or straight ($0° \leq R_{A_1} \leq 90°$)	N/A	More ($t_1 > t_2^*$)	N/A	Turn right
	On right ($-90° \leq R_{A_1} < 0°$)	N/A	More ($t_1 > t_2^*$)	N/A	Turn left
1	Any ($-90° \leq R_{A_1} \leq 90°$)	Not parallel ($R_{H_1} \neq \{180,\ 0\}$)	Less ($t_1 < t_2$)	Not behind both ($t_1 \geq 0$ OR $t_2 \geq 0$)	Go in front
	Any ($-90° \leq R_{A_1} \leq 90°$)	Not parallel ($R_{H_1} \neq \{180,\ 0\}$)	More ($t_1 > t_2$)	Not behind both ($t_1 \geq 0$ OR $t_2 \geq 0$)	Go behind
2	On left ($R_{A_1} > 0°$)	Going left away ($R_{H_1} < 90°$)	Any	Behind both ($t_1 < 0$ AND $t_2 < 0$)	Turn right
	On right ($R_{A_1} < 0°$)	Going right away ($R_{H_1} > 270°$)	Any	Behind both ($t_1 < 0$ AND $t_2 < 0$)	Turn left
3	On left ($R_{A_1} > 0°$)	Going right ($180° < R_{H1} < 360°$)	Equal ($t_1 = t_2$)	In front of both ($t_1 > 0$ AND $t_2 > 0$)	Turn right
	On right ($R_{A_1} < 0°$)	Going left ($0° < R_{H_1} < 180°$)	Equal ($t_1 = t_2$)	In front of both ($t_1 > 0$ AND $t_2 > 0$)	Turn right
4	On left or straight ($R_{A_1} \geq 0°$)	Head-on or same ($R_{H_1} = \{180, 0\}$)	Equal ($t_1 = t_2$)	N/A	Turn right
	On right ($R_{A_1} < 0°$)	Head-on or same ($R_{H_1} = \{180,\ 0\}$)	Equal ($t_1 = t_2$)	N/A	Turn left

Table 4. Summary of collision avoidance logic.

action is either go in front or go behind, the corresponding FIS is activated to execute the maneuver. Therefore, two FISs were developed for this study: go in front and go behind. If a turn left or turn right command is selected, the turn rate of the vehicle will always be a constant value, either positive or negative, depending on which way it should turn. This constant value is one half of the maximum turn rate for collision avoidance maneuvers and one eighth of the maximum turn rate for separation assurance maneuvers.

Both the go in front and go behind fuzzy systems are of Mamdani-type and were constructed in such a way that the input-output relationship can be described using a simple mathematical representation. By using a hybrid representation, as described in Section 2.1, the fuzzy system can easily be expressed mathematically. In this study, the fuzzy systems have a common architecture: triangular membership functions, normalized inputs and outputs, membership function partitioning, product "and" method, minimum implication method, sum aggregation, and mean of maximum defuzzification. If more than one membership function exists for a particular input or output, membership functions are partitioned such that the endpoints of one membership function coincides with the center points of the neighboring membership functions.

Each FIS was developed using a three-input one-output structure. Each FIS uses the distance separating the two aircraft, their relative heading, and their closure rate as inputs to determine the appropriate turn rate output. Since a heterogeneous system is used in this study, the FIS must provide a sufficient turn rate output to avoid a collision for all vehicle type combinations. By considering the separation and closure rate, the conflict can be solved without expelling more energy than required.

In order to use the FISs for both the separation assurance and the collision avoidance platforms, all inputs and outputs must be normalized. Regardless of the avoidance platform being used, the relative heading and closure rate inputs are always normalized by the same values. The relative heading which falls between 0° and 360° is divided by 360°. Thus, a normalized relative heading of 0.5 would represent a head-on encounter. The closure rate is normalized by dividing the true closure rate by the maximum possible closure rate between two vehicles (i.e., 61.762 m/sec in this study). This maximum closure rate would be a result of two fixed wing vehicles approaching one another in a head-on scenario.

The third input, distance, is normalized by 0.4 nmi for separation assurance cases and 0.1 nmi for collision avoidance cases. Lastly, the turn rate output is always between -1 and 1. This output is then scaled by the respective vehicle platform's maximum turn rate. In the case of the collision avoidance system, the output is also multiplied by 1.58. This is to compensate for the fuzzy output not providing a sufficient turn rate command to avoid a collision, especially in head-on scenarios.

Figure 8 shows the structure of each avoidance FIS. Here, the number of membership functions and the corresponding classification can be seen for each input and output. Both the relative heading and the turn rate are partitioned such that the endpoints of the center membership functions coincide with the center points of the adjacent membership functions. As a result of this, and using a product "and" connector, for all possible inputs, at most two of the

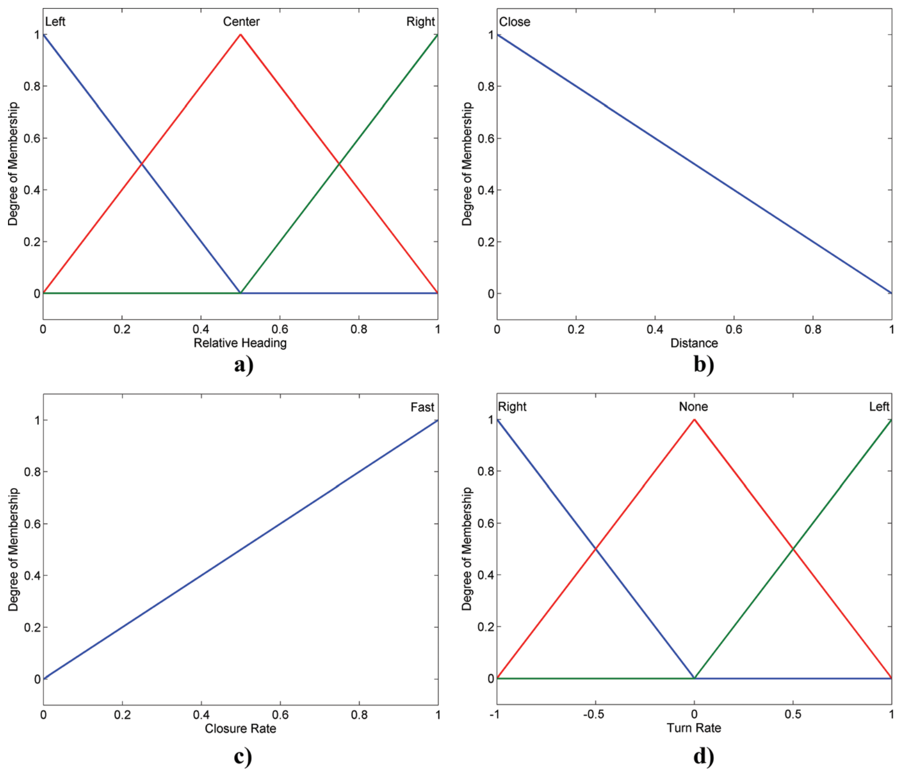

Figure 8. Avoidance FIS structure: (a) Input 1, (b) Input 2, (c) Input 3, (d) Output.

three rules will be active at one time. This drastically reduces the possible solution space, making it much easier to represent the system mathematically.

The input and output membership function sets for both the go behind and go in front FISs are identical. However, the rule bases associating the inputs and outputs are opposite, therefore, when one FIS outputs "turn right" (i.e., negative turn rate), the other FIS would output "turn left" (i.e., positive turn rate), and vice versa. In **Table 5**, the respective rule bases can be seen. Because the other two inputs only have one membership function, they have been excluded from the table.

Rule #	**Input 1: relative heading**	**Go in front output**	**Go behind output**
1	Left	Left	Right
2	Center	Center	Center
3	Right	Right	Left

Table 5. FIS logic.

Mode #	$\mathcal{D}_1$	$\mathcal{D}_2$	$\mathcal{D}_3$
1	[0,0.5)	[0,1]	[0,1]
2	[0.5]	[0,1]	[0,1]
3	(0.5,1]	[0,1]	[0,1]

Table 6. Input domains.

To map the fuzzy system to a set of nonlinear expressions, the following process was used. First, the system was discretized into three possible modes based on the domain of the i^{th} input, D_i. These modes are described in **Table 6**. Here, let a square bracket represent an inclusive bound and a round bracket represent an exclusive bound. Based on the structure of the FIS, if input 1 is exactly 0.5 (Mode 2 in **Table 6**), the output will yield a turn rate of 0. It remains to find the explicit input-to-output mappings for modes 1 and 3.

When the FIS is in mode 1, only the "Left" and "Center" membership functions are active. Thus, the output will be independent of rule 3. Therefore, given that a product "and" method and a mean of maximum aggregation technique is used, the following relationship describes how the output is calculated.

$$\dot{\Psi} = \left[\left(\prod_{i=1}^{3} h_{i_1} \right) + \quad 1 - \prod_{i=1}^{3} h_{i_2} \right) \Bigg] /2 \tag{4}$$

where, $\dot{\Psi}$, is the turn rate, h_{i_1} is the degree of membership for the i^{th} input membership function when using rule 1 (i.e., left, close, fast membership functions), and h_{i_2} is the degree of membership for each input when using rule 2. When substituting the equation of each membership function into Eq. (4), the expression shown in Eq. (5) is found, which can be reduced to the polynomial shown in Eq. (6).

$$\dot{\Psi} = \{[(-2\mu_1 + 1)(-\mu_2 + 1)(\mu_3)] + [1 - (2\mu_1)(-\mu_2 + 1)(\mu_3)]\}/2 \tag{5}$$

$$\dot{\Psi} = (4\mu_1\mu_2\mu_3 - 4\mu_1\mu_3 - \mu_2\mu_3 + \mu_3 + 1)/2 \tag{6}$$

where μ_1, μ_2, and μ_3 are inputs 1, 2, and 3, respectively. The polynomial shown in Eq. (6) can be used to map any combination of inputs that belong to D_1 to an output for the go in front FIS.

Using the same methodology for mode 3, Eqs. (7) through (9) can be found.

$$\dot{\Psi} = \left[\left(-\prod_{i=1}^{3} h_{i_3} \right) + \quad \prod_{i=1}^{3} h_{i_2} - 1 \right) \Bigg] /2 \tag{7}$$

$$\dot{\Psi} = \{[-(2\mu_1 + 1)(-\mu_2 + 1)(\mu_3)] + [(-2\mu_1 + 2)(-\mu_2 + 1)(\mu_3) - 1]\}/2 \tag{8}$$

$$\dot{\Psi} = (4\mu_1\mu_2\mu_3 - 4\mu_1\mu_3 - 3\mu_2\mu_3 + 3\mu_3 - 1)/2 \tag{9}$$

The polynomial shown in Eq. (9) will map any combination of inputs belonging to D_3 to an output for the go in front FIS.

This same approach was used to map any combination of inputs for the go behind FIS to an output using a polynomial function. Since the rule bases are exactly opposite to one another, the output is the negation of Eqs. (6) and (9). Eqs. (10) and (11) describe the input-output relationships for modes 1 and 3, respectively, for the go behind FIS.

$$\dot{\Psi} = \left(-4\mu_1\mu_2\mu_3 + 4\mu_1\mu_3 + \mu_2\mu_3 - \mu_3 - 1\right)/2 \tag{10}$$

$$\dot{\Psi} = \left(-4\mu_1\mu_2\mu_3 + 4\mu_1\mu_3 + 3\mu_2\mu_3 - 3\mu_3 + 1\right)/2 \tag{11}$$

It is important to note that if three or more vehicles are in conflict, each pair of vehicles will be evaluated separately. Thus, for a single ownship, several turn rate outputs will be obtained. Once all respective turn rates are calculated for each intruder, the values are averaged into a single value. This mean turn rate serves as the final vehicle turn rate command.

4. Avoidance algorithm verification

To verify that the avoidance algorithm performed as intended, two levels of avoidance control were evaluated. The first was the high-level, decision-making logic that determined which action a vehicle would take during potential collision scenarios as shown in **Table 4**. The second that was evaluated was the low-level logic in the avoidance FLCs. These FLCs determine the actual vehicle turn rate output after being selected by the decision-making heuristics. For each of these levels, specifications about their behavior were developed and then translated into FOL sentences that could be evaluated by an SMT solver. For the first cases that deal with the avoidance decision-making logic, the specifications and system model were implemented in JKind, with Z3 being used as the SMT solver. The evaluation of the avoidance FISs was performed directly in Z3. The difference was purely a practical one, as the language JKind uses, Lustre [22], is more conducive to easily create a more detailed environment model.

4.1. Collision avoidance decision-making logic

As shown in **Table 4**, there are several conditions for which there are different output actions for avoidance. The specifications were first translated into FOL sentences. The sentences are shown in the following equations. Note that *RAL* is a predicate that acts on the relative angle limit to represent the vehicle's sensors detecting an intruder. Since they can only sense intruders between $\pm 90°$, the relative angle values were limited in the specifications such that being in that range implied that the output action would be the correct one. If the intruder is outside of this range, the specifications do not say what the output action should be. Note that the variables *out*1 and *out*2 are the output actions for the two vehicles. Since both vehicles pick actions based on the states and have no knowledge of the other vehicle's selected action, the specifications need to go both ways. Also, the actions are represented by integer values 1–4 such that {1,2,3,4} = {go behind, go in front, turn left, turn right}, respectively.

Although several avoidance actions could be taken by each vehicle, only cases 1, 3, and 4 required formal verification. In these cases, both vehicles can sense one another and will perform some type of avoidance maneuver. Thus, their actions must be such that they do not move closer to one another. In cases 0 and 2, however, there was no need to write requirements because only one vehicle performs an avoidance action. In case 0, vehicle 2 is stationary, so it does not try to avoid vehicle 1. In case 2, vehicle 2 cannot sense vehicle 1, thus, there is no need to check that the two vehicles' actions produce a converging result.

Eqs. (12) through (15) are the specifications for case 1 in **Table 4** and are intended to ensure that the vehicles do not both choose to go in front, or go behind.

$$S_{1_1} = \forall R_{A_1} \forall R_{A_2} \forall R_{H_1} \forall R_{H_2} (RAL \rightarrow (out1 = 1 \rightarrow out2 = 2)) \tag{12}$$

$$S_{1_2} = \forall R_{A_1} \forall R_{A_2} \forall R_{H_1} \forall R_{H_2} (RAL \rightarrow (out1 = 2 \rightarrow out2 = 1)) \tag{13}$$

$$S_{1_3} = \forall R_{A_1} \forall R_{A_2} \forall R_{H_1} \forall R_{H_2} (RAL \rightarrow (out2 = 1 \rightarrow out1 = 2)) \tag{14}$$

$$S_{1_4} = \forall R_{A_1} \forall R_{A_2} \forall R_{H_1} \forall R_{H_2} (RAL \rightarrow (out2 = 2 \rightarrow out1 = 1)) \tag{15}$$

Eqs. (16) and (17) are the specifications for case 3 and ensure that the vehicles turn the same way when resolving these particular conflicts. Recall that in this case, both vehicles are at the same time from the crossing point. This specification ensures that they will then be forced into new positions such that the crossing times are not equal and the FISs are then selected.

$$\begin{aligned} S_{3_1} = {} & \forall R_{A_1} \forall R_{A_2} \forall R_{H_1} \forall R_{H_2} (RAL \rightarrow ((t_1 = t_2 \wedge t_1 > 0 \wedge t_2 > 0 \wedge ((R_{H_1} \geq 0 \wedge R_{H_1} \\ & < 180) \vee (R_{H_1} > 180 \wedge R_{H_1} < 360))) \rightarrow (out1 = 4 \wedge out2 = 4))) \end{aligned} \tag{16}$$

$$\begin{aligned} S_{3_2} = {} & \forall R_{A_1} \forall R_{A_2} \forall R_{H_1} \forall R_{H_2} (RAL \rightarrow ((t_1 = t_2 \wedge t_1 > 0 \wedge t_2 > 0 \wedge ((R_{H_2} \geq 0 \wedge R_{H_2} \\ & < 180) \vee (R_{H_2} > 180 \wedge R_{H_2} < 360))) \rightarrow (out2 = 4 \wedge out1 = 4))) \end{aligned} \tag{17}$$

Finally, Eqs. (18) and (19) are for case 4. These two specifications are for cases where the vehicles are head-on, or traveling next to each other in the same direction, respectively. The first specification ensures that while in a head-on encounter, both vehicles turn the same direction (i.e., both left, or both right, forcing them to diverge). The second specification ensures that while traveling in the same direction, the vehicles turn in opposite directions.

$$\begin{aligned} S_{4_1} = {} & \forall R_{A_1} \forall R_{A_2} \forall R_{H_1} \forall R_{H_2} (RAL \rightarrow ((t_1 = t_2 \wedge (R_{H_1} = 180 \wedge R_{H_2} \\ & = 180)) \rightarrow ((out1 = 3 \wedge out2 = 3) \vee (out1 = 4 \wedge out2 = 4)))) \end{aligned} \tag{18}$$

$$\begin{aligned} S_{4_2} = {} & \forall R_{A_1} \forall R_{A_2} \forall R_{H_1} \forall R_{H_2} (RAL \rightarrow ((t_1 = t_2 \wedge (R_{H_1} = 0 \wedge R_{H_2} \\ & = 0)) \rightarrow ((out1 = 3 \wedge out2 = 4) \vee (out1 = 4 \wedge out2 = 3)))) \end{aligned} \tag{19}$$

where t_1 is the time until the crossing point, R_{H_1} is the relative heading from vehicle 1 to vehicle 2, and R_{A_1} is the relative angle from vehicle 1 to vehicle 2. Similarly, t_2, R_{H_2}, and R_{A_2} are for vehicle 2. In addition to these specifications, another specification was created to ensure that the vehicles always chose one of the valid actions. This specification is not shown but is similar to Eqs. (12) through (19) such that the output actions are one of the possible outcomes (1–4). These specifications were then translated to a temporal representation for evaluation in JKind using previously developed methods [18].

4.2. Avoidance fuzzy inference systems

The method for verifying the avoidance FISs is similar, but the specifications were left in FOL and then implemented directly into Z3. The main reason for this was that less of the system model was needed to check these outputs. The behavior that the specifications needed to encapsulate was that the turn rate output for each FIS needs to always be in the correct direction. The correct direction for each of these cases is shown in **Table 7**.

These can then be encoded in FOL sentences using the polynomial representation of the FISs shown in Section 3.4. These sentences are then negated to show that there are no possible variable values that make the negated sentences true. The negated FOL sentences are shown in Eqs. (20) through (23). Note that μ_1, μ_2, μ_3, $\dot{\Psi}$, and modes 1 and 3 are the same as detailed in Section 3.4.

$$S_{behind_1} = \exists\mu_1\exists\mu_2\exists\mu_3\left(\dot{\Psi}\left(\mu_1,\mu_2,\mu_3\right) \geq 0\right) \tag{20}$$

$$S_{behind_3} = \exists\mu_1\exists\mu_2\exists\mu_3\left(\dot{\Psi}\left(\mu_1,\mu_2,\mu_3\right) \leq 0\right) \tag{21}$$

$$S_{front_1} = \exists\mu_1\exists\mu_2\exists\mu_3\left(\dot{\Psi}\left(\mu_1,\mu_2,\mu_3\right) \leq 0\right) \tag{22}$$

$$S_{front_3} = \exists\mu_1\exists\mu_2\exists\mu_3\left(\dot{\Psi}\left(\mu_1,\mu_2,\mu_3\right) \geq 0\right) \tag{23}$$

These negated sentences were then implemented in Z3. If Z3 finds that these sentences were all unsatisfiable, there are no possible real-valued assignments to μ_1, μ_2, or μ_3 that allow the FISs to turn the vehicles in an undesired direction.

Mode	Action	Intruder position	Intruder direction	Turn direction
1	Go behind	On right	Going left	Right (negative)
3	Go behind	On left	Going right	Left (positive)
1	Go in front	On right	Going left	Left (positive)
3	Go in front	On left	Going right	Right (negative)

Table 7. Avoidance FIS outputs.

5. Simulation environment descriptions

5.1. Testing environment

Prior to integrating the controllers presented in Section 3 into the full simulation environment, each avoidance platform was tested using pairwise encounter scenarios. This component testing was used to ensure that each was operating as desired. In this section, the methods used to test each controller are described.

5.1.1. Separation assurance

A testing environment was created to evaluate a considerable amount of pairwise encounters between aircraft. This testing environment was used to identify potential controller failures. To accomplish this, various initial relative headings and relative angles were tested. In all cases, the initial location and heading of one aircraft was held constant. Then, by placing the intruding vehicle at a relative angle of −90° and just outside the vehicle's sensing radius, as shown in **Figure 9**, we evaluated the interactions for 720 initial intruder heading values. This was then repeated four more times by changing the initial relative angle between −90° (to the right) and 0° (straight ahead). A visualization of these scenarios is shown in **Figure 9**. Note that the radius of the sensing semi-circle for the separation assurance tests was set at 0.4 nmi.

Although intruders can approach a vehicle from the left, that is, a relative angle between 0° and 90°, symmetry allows us to limit the initial relative angles to lie between −90 and 0°. To verify, a sample scenario was tested for the full field of view. In all cases, the trajectories of the vehicles were symmetric to one another (i.e., when reflected across the vehicle's initial heading).

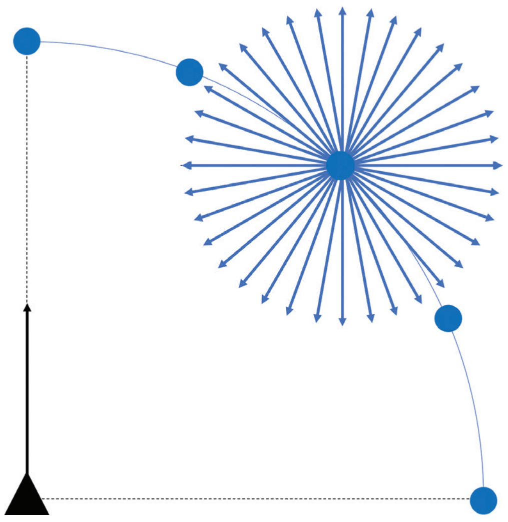

Figure 9. Separation and avoidance testing scenarios between ownship (triangle) and intruder UAS for multiple different initial positions (circles).

5.1.2. Collision avoidance

The collision avoidance platform was tested in the same manner as the separation assurance. The difference being that the sensing radius for the interaction tests as shown in **Figure 9** is 0.1 nmi. Thus, the initial position of each intruder scenario was just outside of this sensing radius.

5.2. Full simulation environment

5.2.1. Airspace description

The simulation environment created for this study models a portion of the US airspace over central Ohio. A depiction of the selected airspace can be seen in **Figure 10**. This airspace covers approximately 2500 square miles where sUAS can operate at a maximum altitude of 400 ft.

5.2.2. Mission types and objectives

Many sUAS will be active throughout this airspace, each with individual missions, such as, precision agriculture, forest monitoring, roadway surveillance, disaster management, and package delivery. During simulation runs, the UAS will travel to various waypoints to fulfill their assigned missions. After visiting all waypoints for a given mission, the aircraft will return to their respective starting locations. For more on the mission types, please refer to Refs. [11, 12].

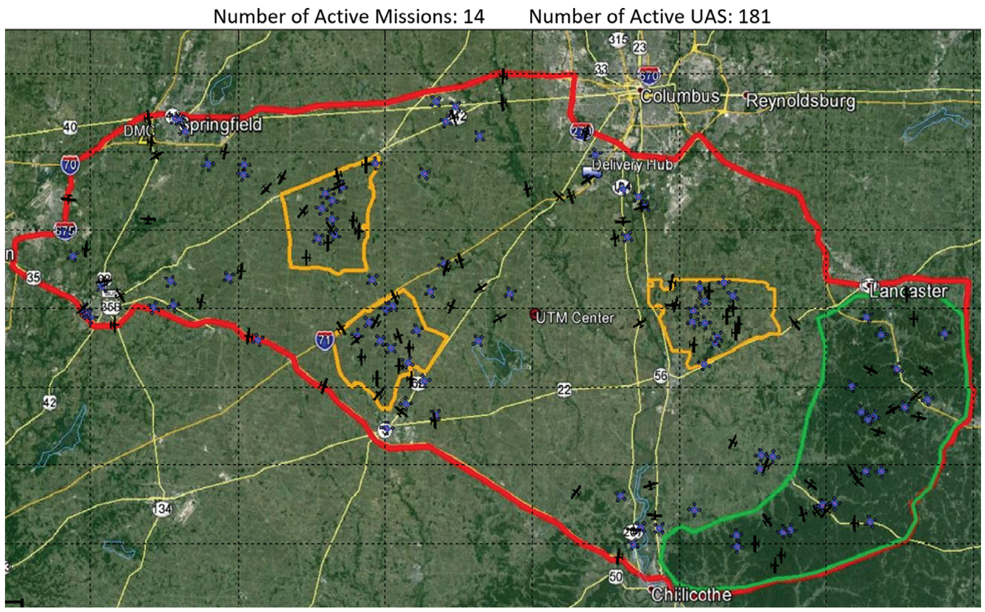

Figure 10. Simulation platform example.

5.2.3. Scalability and operations

The full simulation environment can accommodate any number of sUAS and missions. However, these numbers were constant throughout testing. A maximum of 184 sUAS can be airborne at any given time. This is a result of having centralized control of the aircraft. In a realistic environment, some of the computationally heavy components can be handled in parallel onboard each individual sUAS.

5.2.4. Aircraft models

Two vehicle platforms were used in all simulations: fixed wing and quad-rotor. Kinematic models were developed for each within the constraints of the environment. These constraints include maximum turn rate, maximum climb/decent rate, minimum/maximum speed, and maximum altitude.

For this study, each aircraft is assumed to climb at 4 ft./sec upon takeoff. The vehicles will continue to climb at this rate until an altitude of 400 ft. is reached. When vehicles are in conflict and are required to make trajectory modifications, all adjustments are constrained to lateral deviations in flight path (i.e., speed and altitude modifications are not used). Therefore, UAS are limited to level, two-dimensional flight during conflict resolution scenarios. Using this assumption, the maximum turn rate for each fixed wing vehicle is described in Eq. (24).

$$\dot{\Psi}_{max} = \frac{g\sqrt{n^2 - 1}}{V} \tag{24}$$

All fixed wing vehicles travel at 60 knots and have a maximum load factor of 3.5. Therefore, they are constrained to a maximum turn rate of 61.06 deg/sec. The multi-rotor systems, however, can travel at a maximum airspeed of 38 knots. Due to the nature of the quad-rotor sUAS, Eq. (24) does not accurately model the maximum turn rate for this vehicle type and it is assumed that the aircraft can yaw at a maximum rate of 45 deg/sec. Also, each aircraft is assumed to have the capability to detect an intruding aircraft at a distance of 0.1 nmi if within a 180° field of view in front of the aircraft (i.e., $-90° \leq R_{A_1} \leq 90°$).

As previously described in **Table 1**, the collision threshold is defined by aircraft coming within 60 m laterally and 50 ft. vertically of one other. To track the vehicles, ground-based sensors for detecting aircraft have been dispersed in the airspace. These sensors, along with telemetry data, provide continuous and reliable vehicle state information (i.e., position and velocity). This capability allows the implementation of global separation assurance practices. More details about the simulation environment and its properties can be found in Refs. [11, 12]. (NOTE: The collision buffer value was used to accommodate for position uncertainties while tracking the sUAS. In this study, the avoidance system has perfect knowledge of all vehicle state information acquired from ground based and onboard sensors.)

6. Results

6.1. Avoidance systems testing

For each pairwise encounter shown in **Figure 9**, the closest point of approach (CPA) was found. The CPA is defined as the minimum recorded distance between the two vehicles throughout the entire encounter. For both the collision avoidance and separation assurance platforms, a total of four different vehicle platform trials were conducted: fixed vs. fixed, quad vs. quad, quad vs. fixed, and fixed vs. quad. The first vehicle type designation represents the ownship vehicle's type in **Figure 9** (i.e., has the same starting position and heading for all tested cases), whereas, the second vehicle platform designation represents the intruder vehicle's type (i.e., initial conditions change for each tested scenario).

To measure the effectiveness of the avoidance logic, the minimum CPA was recorded for all initial relative angles tested. In addition, the total number of collisions for each case were tallied, and for the separation assurance case, the total number of LOSs. As a reminder, a collision is deemed by two vehicles coming within 60 m of one another, and an LOS is when two vehicles come with 0.1 nmi of one another.

The results for the separation assurance testing have been shown in **Table 8**. Here, the "angle case" refers to the various initial relative angles, from -90 to $0°$, respectively. Thus, case 1 represents an intruder directly to the right, and case 5 represents an intruder directly in front of the ownship.

Although all vehicle platform combinations had more than one LOS, no LOS resulted in two vehicles colliding (i.e., a CPA less than 60 m). To try and understand where the separation assurance platform was breaking down to allow an LOS to occur, the CPA results were plotted for each initial intruder heading. When evaluating the results of each vehicle case, it was found that although several LOSs occurred, the failures tended to lie in groups near the same intruder angle depending on the initial position of the intruder. For example, in the fixed vs. quad case, it can be seen that no LOSs occurred in the first three trials. However, once the intruder was positioned such that it was nearly in front of or directly in front of the ownship,

Angle case	Fixed vs. Fixed		Quad vs. Quad		Fixed vs. Quad		Quad vs. Fixed	
	Min CPA (m)	LOS	Min CPA (m)	LOS	Min CPA (m)	LOS	Min CPA (m)	LOS
1	185.3	0	185.1	0	574.3	0	185.8	0
2	185.6	0	185.5	0	351.4	0	158.6	12
3	185.3	0	185.4	0	185.6	0	133.2	24
4	186.1	0	131.2	9	175.1	24	137.3	31
5	98.4	116	137.0	8	128.1	161	132.0	28

Table 8. Separation assurance testing results.

the separation assurance platform began to fail. This was caused by the fixed wing vehicle traveling at higher speeds than the quad-rotor vehicle. Therefore, when the intruder heading was away from the ownship, the ownship tended to approach the intruder from behind and begin to pass the vehicle. When approaching the vehicle from behind, it proved difficult for the avoidance platform to solve the conflict prior to an LOS in all vehicle configurations, as shown by angle case 5.

Table 9 shows the results of the collision avoidance testing. Each case had a minimum CPA greater than 60 m, implying no collisions were found throughout testing. Although not all possible encounter scenarios have been tested, this shows that the avoidance system logic is quite robust. As seen from the results, the homogeneous quad-rotor and the fixed vs. quad cases showed promising results. They consistently had a higher minimum CPA than the other two cases. The closure rates in the fixed vs. fixed cases were higher than cases involving quad-rotors. This resulted in consistently lower minimum CPA values. A noteworthy result is in the fixed vs. fixed scenario for angle case 5. The intruder being head-on and directly in front of the ownship resulted in a CPA of 60.8 m. Although this is close to the collision boundary, this shows that even in the highest closure rate scenario, the collision avoidance system was able to resolve the conflict.

6.2. Formal verification

6.2.1. Avoidance logic

After evaluating all the specifications outlined in Eqs. (12) through (19), JKind returned that they always held. This means that for all possible real-valued assignments to the variables (within the sensor domain limitations), the vehicles will always select the desired output action.

Although the final version of the avoidance logic adhered to all of the specifications, during development there were several cases where JKind found values that violated one or more of the specifications. These counterexamples are invaluable as they identify exact cases that result in undesired behavior. This gives way to corrections based on the counterexample conditions.

	Fixed vs. Fixed	**Quad vs. Quad**	**Fixed vs. Quad**	**Quad vs. Fixed**
Angle Case	**Min CPA (m)**	**Min CPA (m)**	**Min CPA (m)**	**Min CPA (m)**
1	132.1	138.9	154.7	125.3
2	111.9	125.2	133.3	121.0
3	101.9	113.5	122.9	111.1
4	82.9	104.4	103.5	97.8
5	60.8	96.7	98.9	99.0

Table 9. Collision avoidance testing results.

Specification	R_{H_1}	R_{A_1}	$out1$	R_{H_2}	R_{A_2}	$out2$
S_{3_2}	180	$-5 \cdot 10^{-15}$	3	180	$-5 \cdot 10^{-15}$	4

Table 10. Avoidance logic counter-example values.

As an example of this, one of the conditions that violated a specification found during development is shown in **Table 10**.

These conditions mean that one vehicle is heading in the exact opposite direction of the other and there is a slight position offset between them as shown in **Figure 11**. One vehicle selects the turn left action (3) while the other selects the turn right action (4). This implies they are not turning away from each other. The reason for this was that the range of angles that would force vehicle 1 into the correct action was not inclusive on one of its boundaries. This then meant that the conditions forced vehicle 1 into a different action and generated this counterexample.

6.2.2. *Avoidance FISs*

Similarly, after evaluating the specifications in Eqs. (20) through (23), Z3 showed that they were all unsatisfiable. This shows that the FLCs will always make the ownship turn away from an intruder.

6.3. Full simulation results

To test the algorithms in a dynamic environment, simulations were run both with the separation assurance mitigations and then without. The number of LOSs and number of collisions were recorded in order to directly compare the mitigated and unmitigated cases.

6.3.1. *Unmitigated study*

For this unmitigated study, the separation assurance system was disabled. The results of this study are shown in **Table 11**.

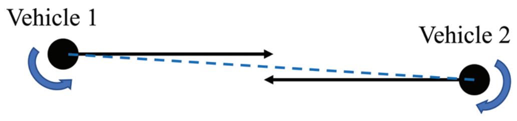

Figure 11. Counterexample showing head-on vehicles turning into each other.

Separation less than 0.4 nmi	LOSs	Collision avoidance maneuvers	Collisions	Collision avoidance success rate	Number of flight hours	LOSs per flight hour	Collisions per flight hour
26,576	8263	8252	2	99.98%	25,116	0.33	$7.96 \cdot 10^{-15}$

Table 11. Results for simulation without separation assurance.

Over the span of 25,116 flight hours, there were 26,576 recorded violations of the 0.4 nmi separation threshold. These resulted in 8263 LOSs. The collision avoidance algorithm was employed for all except for 11 LOS occurrences. In those cases, the vehicles were outside of one another's field of view, thus the collision avoidance system was not used. The collision avoidance system was 99.98% successful at resolving conflicts.

The only collisions that occurred throughout simulation can be attributed to the restriction on the detection sensor field of view and having no memory of state time histories. Therefore, if two vehicles were nearly parallel and directly beside one another, they would turn to resolve the conflict (i.e., turn away from one another). However, this turning again puts each intruder outside of the other vehicle's field of view. The lack of state memory combined with no sensor input caused a switch back to their navigation controllers. The navigation controller caused them to go back toward one another. Since the navigation controller had a higher turn rate output than the avoidance output, this cycle would continue (each vehicle turning away then toward) until they converged and were within 60 m of one another.

6.3.2. Mitigated study

For this study, the separation assurance features were enabled to help mitigate the risk of having an LOS. Although fewer LOSs were expected with the mitigations enabled, some LOSs were expected due to sub-optimal performance in head-on and trailing situations. The results of this mitigated study are shown in **Tables 12** and **13**.

Table 12 presents the results of the separation assurance platform. In this mitigated study, only 9277 flight hours were recorded. Thus, sUAS were able to complete their respective missions in a shorter period of time. Throughout the simulation aircraft came within 0.4 nmi on 33,550 occasions. However, of those instances, the separation assurance system predicted an LOS to occur within 2 min only 14,750 times. Of these resolution advisories, only 75.74% were successful, resulting in 3579 LOSs and 0.39 LOSs per flight hour. Although this number is slightly larger than the number of LOSs per flight hour in the unmitigated study, this additional layer of avoidance kept vehicles from entering any scenarios that resulted in collision. Thus, the overall safety of the UTM system has been improved.

Separation less than 0.4 nmi	Separation assurance maneuvers	LOSs	Separation assurance success rate	Number of flight hours	LOSs per flight hour
33,550	14,750	3579	75.74%	9277	0.39

Table 12. Separation assurance results.

Collision avoidance maneuvers	Collisions	Collision avoidance success rate	Number of flight hours	Collisions per flight hour
3568	0	100.00%	9277	0

Table 13. Collision avoidance results.

If an LOS occurred and the vehicles were within one another's sensor ranges, the sense and avoid software would activate. The results of the collision avoidance software can be seen in **Table 13**. Of the 3568 encounters, the collision avoidance software was 100.00% successful at resolving conflicts.

7. Conclusion

In this work, multiple fuzzy logic controllers and decision-making systems were used in conjunction to prevent potential losses of separation in a congested, three-dimensional airspace. This simulation environment allowed for extensive encounter scenarios between heterogeneous vehicles to test the two conflict resolution systems. First, a sense and avoid system was developed to prevent potential collisions using only current state information and without communication between vehicles. Next, a separation assurance platform was developed to further mitigate the risk of a potential collision. This platform uses global aircraft state information to predict if two aircraft will have an LOS within a given look-ahead time. If an LOS was predicted, the system would issue necessary resolution advisories to the proper aircraft to prevent an LOS.

Once the controllers were developed, numerical simulations and formal methods were used to verify the controllers performed as expected. Using a formal methods approach, we could show that the controller output was always in the correct direction (i.e., always performed as expected). In addition, we were able to verify that in all pairwise encounter scenarios between sUAS, the actions of each vehicle were such that they would never turn toward one another when avoiding a collision.

After a formal methods approach verified the control logic behavior and fuzzy logic controller outputs, numerical simulations were conducted. In all simulations, the avoidance system had perfect knowledge of all vehicle state information (i.e., speed, heading, and location). For the collision avoidance scenarios tested, the fuzzy system was successful at resolving all potential conflicts for both the homogeneous and heterogeneous cases. However, the separation assurance platform had trouble resolving certain types of encounter scenarios. Thus, it sometimes would not prevent an LOS between vehicles. However, when an LOS occurred, the collision avoidance system again prevented any mid-air collisions from occurring.

Several full simulation environment missions were also run to evaluate the effectiveness of the avoidance algorithms. These missions included cases where the separation assurance mitigations were both enabled and disabled. Overall, the results of this experiment were as expected. In the mitigated study, no collisions between aircraft occurred. However, when the mitigations were removed, vehicles encountered scenarios where the collision avoidance system could not prevent a collision. These collisions were not due to the collision avoidance logic or the fuzzy logic controllers, but were attributed to the limited vehicle sensor performance and lack of memory.

For future work, we aim to improve upon the separation assurance techniques to prevent an LOS in all encounter scenarios. Also, representing the system with a higher fidelity model of the environment in the formal methods tools would allow for more complete specifications (i.e., vehicles never lose separation) and then identify cases that violate them. In addition, since the avoidance system had perfect vehicle state information, we would like to introduce a level of uncertainty to the sensor models. Finally, we wish to implement the proposed avoidance software into hardware testing environments.

Appendix: Nomenclature

CPA	Closest Point of Approach
FIS	Fuzzy Inference System
FLC	Fuzzy Logic Controller
FOL	First Order Logic
LOS	Loss of Separation
NAS	National Airspace System
pCPA	Predicted Closest Point of Approach
SAT	Satisfiability (Boolean)
SMT	Satisfiability Modulo Theories
sUAS	Small Unmanned Aerial System
tLOS	Time to Loss of Separation
UAS	Unmanned Aerial System
UTM	UAS Traffic Management

Author details

Brandon Cook[1,2]*, Tim Arnett[2] and Kelly Cohen[2]

*Address all correspondence to: cookb9@mail.uc.edu

1 NASA Ames Research Center, Moffett Field, CA, USA

2 Department of Aerospace Engineering and Engineering Mechanics, University of Cincinnati, Cincinnati, OH, USA

References

[1] Cook K. The silent force multiplier: The history and role of UAVs in warfare. Proceedings of the IEEE Aerospace Conference; 3–10 March 2007; Big Sky, MT, USA. IEEE. 2007:1-7. DOI: 10.1109/AERO.2007.352737

[2] Ollero A, Merino L. Unmanned aerial vehicles as tools for forest-fire fighting. Forest Ecology and Management. 2006;**234**(1):S263

[3] Bamburry D. Drones: Designed for product delivery. Design Management Review. 2015;**26**(1):40-48. DOI: 10.1111/drev.10313

[4] Prevot T, Homola J, Mercer J. Human-in-the-loop evaluation of ground-based automated separation assurance for NEXTGEN. In: The 26th Congress of ICAS and 8th AIAA ATIO. 2008 Sep. 8885

[5] Erzberger H, Heere K. Algorithm and operational concept for resolving short-range conflicts. Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering. 2010 Feb 1;**224**(2):225-243

[6] Lauderdale TA, Erzberger H. Automated separation assurance with weather and uncertainty. Air Traffic Management and Systems. 2014:35-47. Springer, Japan

[7] Temizer S, Kochenderfer M, Kaelbling L, Lozano-Pérez T, Kuchar J. Collision avoidance for unmanned aircraft using Markov decision processes. In: AIAA Guidance, Navigation, and Control Conference. 2010 Aug 2. 8040

[8] Durand N, Alliot JM, Noailles J. Collision avoidance using neural networks learned by genetic algorithms. In: IEA-AEI 1996, 9th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert systems. 1996 Jun 1

[9] Hromatka M. A fuzzy logic approach to collision avoidance in smart UAVs. Honors Thesis. Collegeville, MN: College of Saint Benedict and Saint John's University; 2013

[10] Kuchar JE, Drumm AC. The traffic alert and collision avoidance system. Lincoln Laboratory Journal. 2007 Nov 2;**16**(2):277

[11] Cook B, Cohen K, Kivelevitch EH. A fuzzy logic approach for low altitude uas traffic management (UTM). In: AIAA Infotech@ Aerospace. 2016:1905. DOI: 10.2514/6.2016-1905

[12] Cook BM. Multi-Agent Control Using Fuzzy Logic. Electronic Thesis or Dissertation. University of Cincinnati. 2015. OhioLINK Electronic Theses and Dissertations Center. pp. 90-166

[13] Peled D. Software Reliability Methods. New York, NY: Springer Science & Business Media; 2013. 332

[14] Baier C, Katoen J, Larsen K. Principles of Model Checking. Cambridge, MA: MIT press; 2008. 984

[15] Clark M, Rattan K. Piecewise affine hybrid automata representation of a multistage fuzzy pid controller. In: Proceedings of the 2014 AAAI Spring Symposium Series: Formal Verification and Modeling in Human-Machine Systems; 24–26 March 2014

[16] Ross T. Fuzzy Logic with Engineering Applications. 3rd ed. Hoboken, NJ: John Wiley & Sons; 2009. 606

[17] Butler RW. “What is Formal Methods?” NASA LaRC Formal Methods Program [Internet]. 2001. Available from: https://shemesh.larc.nasa.gov/fm/fm-what.html. [Accessed: 03-02-2017]

[18] Wagner L, Fifarek A, DaCosta D, Gross K. SpeAR: Specification and Analysis of Requirements. InS5 Symposium. 2014

[19] Hagen GE. Verifying safety properties of Lustre programs: An SMT-based approach. Ann Arbor, MI: ProQuest; 2008

[20] Ghassabani E, Gacek A, Whalen MW. Efficient generation of inductive validity cores for safety properties. In: Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering. 2016 Nov 1:314-325. ACM

[21] De Moura L, Bjørner N. Z3: An efficient SMT solver. In: International Conference on Tools and Algorithms for the Construction and Analysis of Systems. 2008 Mar 29. 337-340. Berlin Heidelberg: Springer

[22] Halbwachs N, Caspi P, Raymond P, Pilaud D. The synchronous data flow programming language LUSTRE. Proceedings of the IEEE. 1991 Sep;**79**(9):1305-1320