A Sense and Avoid System for Unmanned Aircraft in Formation Flight*

Jared K. Cooper,† Nathan D. Richards,‡ John D. Schierman,§ David Neal**
Matthew D. Lichter,†† Thomas Schlapkohl,‡‡ Shelly Su§§
Barron Associates, Inc., Charlottesville, VA, 22901

Abstract

This paper summarizes development of a sense and avoid system (SAA) for a fleet of unmanned aircraft systems (UASs) in formation flight that are threatened by a non-cooperative intruder. The intruder is not transponding its position and is assumed to be unaware of the impending collision. It is assumed the UASs are equipped with electro-optical, forward-looking infrared cameras, or other sensors such as radar to detect the intruder. Low-level sensor processing, detection, and tracking systems are not the focus of this work; however, we do explore the unique sensor fusion and estimation capabilities afforded by a formation of UASs. Using intra-fleet communication and a sensor fusion algorithm, the formation aircraft effectively form a stereo-optic solution by sharing their sensory information, which provides a rapidly-converging estimate of the intruder's position and velocity. At the core of the SAA system is a conflict resolution algorithm that continuously generates and updates a resolution path which satisfies a hierarchical set of safety constraints, including a pre-defined "well clear" volume. If the well clear criterion cannot be achieved, due to late detection of the intruder, for example, the conflict resolution algorithm maximizes the separation distance. The conflict resolution algorithm can generate a lateral avoidance maneuver, a vertical avoidance maneuver, or a combined maneuver, depending on the scenario and a user-defined resolution "strategy". The resolution strategy allows the overall system to adapt to evolving regulations and practices related to UAS incorporation in the NAS and other theaters. The airspeed of the formation can also be adjusted to aid in conflict resolution. The SAA algorithm has the ability to keep the fleet in formation during the conflict avoidance, or depending on the geometry of the scenario, allow the formation to split into sub-formations, each executing a separate avoidance path.

I. Introduction

From the current and recent conflicts in Afghanistan and Iraq, it is clear that unmanned aircraft systems (UASs) are playing an increasing role in critical military missions. Unmanned aerial vehicles have the potential to revolutionize a number of important military, government, and commercial operations. UASs have been proposed to perform a wide variety of tasks including intelligence, surveillance, and reconnaissance (ISR), border patrol, security monitoring, search and rescue, atmospheric research, aerial photography and mapping, and traffic monitoring. As technologies mature, a growing number of UAS implementations are being applied to real-world tasks. While UASs continue to revolutionize military operations, the next frontier will involve a collective of UASs collaborating to perform tasks such as reconnaissance, threat detection, target tracking, and other advanced mission scenarios. These mission scenarios necessitate the development of an advanced motion planner capable of

* This paper has been cleared for public release by the Air Force. Case No: 88ABW-2013-5190
† Research Scientist, 1410 Sachem Place, Ste. 202, Member.
‡ Research Scientist, 1410 Sachem Place, Ste. 202, Member.
§ Principal Research Scientist, 1410 Sachem Place, Ste. 202, Associate Fellow.
** Research Scientist, 1410 Sachem Place, Ste. 202.
†† Senior Research Engineer, 1410 Sachem Place, Ste. 202.
‡‡ Engineer, 1410 Sachem Place, Ste. 202.
§§ Research Engineer, 1410 Sachem Place, Ste. 202.

dynamically generating 3-D routes that facilitate coordinated action. The major challenge to using platforms in these military roles is that they pose the risk of unintentional collision between different UASs or manned aircraft. In critical avoidance scenarios, UASs under autonomous flight do not currently employ the same level of situational awareness and decision making as offered by a human pilot in manned aircraft.

Development of autonomous sense and avoid (SAA) technologies are needed for UAS platforms to be fully integrated with manned aircraft in air war operations or for civilian applications in the National Airspace System (NAS). Initial development of SAA capabilities has been accomplished by a recent AFRL program involving Northrop Grumman and Calspan, flight testing a passive ranging and collision avoidance concept [1,2]. However, further research and development is needed to augment these technologies to address multiple UASs in close formation flight. Ensuring collision-free maneuvers with teams of aircraft presents several unique challenges to system designers. At a coordination level, one must consider that the flow of information among agents may be severely restricted, due to security requirements and/or tight bandwidth limitations. Consequently, it is important to develop coordinated motion strategies that limit the amount of information passed over the network, and perform robustly in the presence of communication failures such as drop outs and unexpected latencies.

Inclusion of UAS platforms in the battle space can involve all classes and sizes (Tier I, II, III). Autonomous SAA systems must utilize varied sensor suites and address heterogeneous classes of intruders. Further, SAA systems should be capable of sensing and detecting cooperative and non-cooperative intruders, and as such, should be capable of using optical, radar, or other passive or active ranging sensors, and Automatic Dependent Surveillance Broadcast (ADS-B) information, which will be required as part of the Next Generation Air Transportation System (NextGen).

The following section outlines the scope of the work presented in this paper and underlying assumptions. Section III summarizes the primary components of the MUSAA system and Section IV details the conflict resolution path planner. Results of various intruder scenarios are illustrated in Section V, with embedded hardware test results summarized in Section VI. Finally, the paper is completed with conclusions and recommendations for future work I n Section VII.

II. Overview of Program Scope

Our primary focus in this work is to develop a SAA system for a fleet of UASs in formation flight that is threatened by a non-cooperative intruder. The intruder is not transponding its position and is assumed to be unaware of the impending collision. The intruder is assumed to approach the fleet from any forward angle (from 110 deg. left to 110 deg. right) and from above, below or at the same altitude. The SAA system can treat simultaneous intruders. The case of an intruder overtaking the formation from behind has not been addressed primarily due to the responsibilities of the ownship and intruder in an overtaking scenario per 14 CFR 91.113: an overtaking vehicle must detect the conflict and execute the avoidance maneuver.

It is assumed the UASs are equipped with electro-optical (EO), forward looking infrared (IR) cameras, radar, or other sensors to detect the intruder. Low-level sensor processing, detection, and tracking systems are not the focus of this work. Instead, we focus on developing a sensor fusion algorithm that uses shared information from the fleet's EO sensors, exploiting the multiple perspectives offered by the formation and forming a stereo-optic solution to estimate the intruder's position and velocity. The sensor fusion algorithm is discussed in detail in [3].

Nominally, the developed SAA algorithm attempts to keep the fleet in formation during the conflict resolution. However, depending on the geometry of the scenario and capabilities of the aircraft, the SAA algorithm can allow the formation to split into sub-formations – each executing a separate avoidance path – if this aids in achieving the hierarchical safety constraints. These scenarios are depicted in Figure 1. This article presents development and results with the formation maintained and a future article will discuss formation splitting and rejoining.

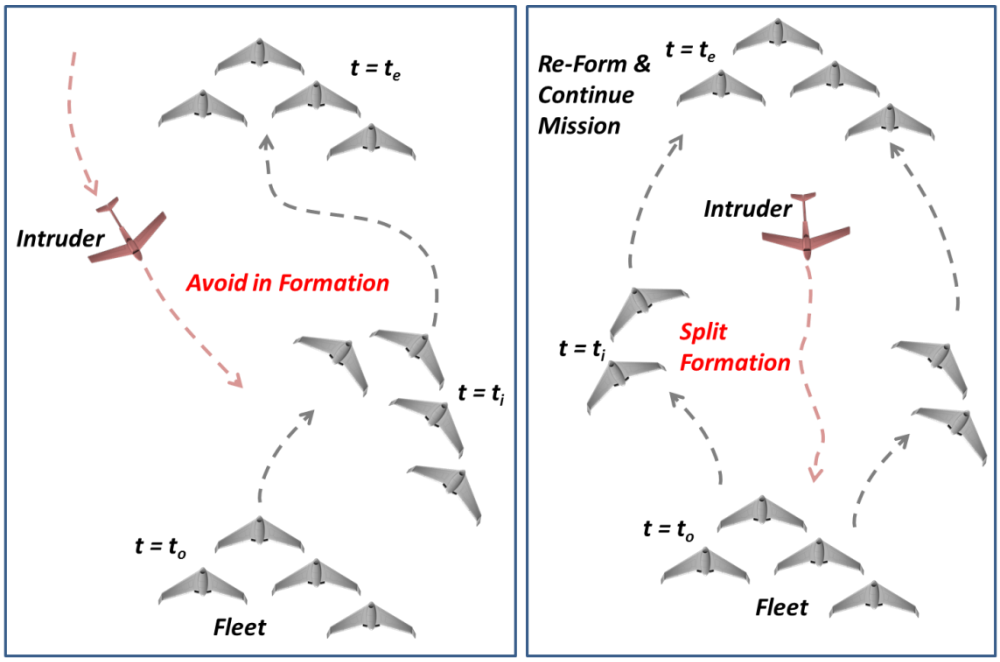

Figure 1. Avoidance Scenarios Studied

III. Overview of Developed Technologies

A. Architecture Overview

We denote the developed SAA system as Multi-Vehicle Unmanned Aircraft Systems Sense And Avoid (MUSAA). The MUSAA algorithms have been developed with the ultimate goal of flight testing and hence the underlying architecture and support structure reflect real-world flight testing environments. A software integration laboratory has been constructed for initial soft-real-time testing of core elements in the MUSAA system. The MUSAA architecture is comprised of simulated ground-based software and on-board software located on each formation aircraft. Figure 2 represents a high level schematic of the overall MUSAA architecture.

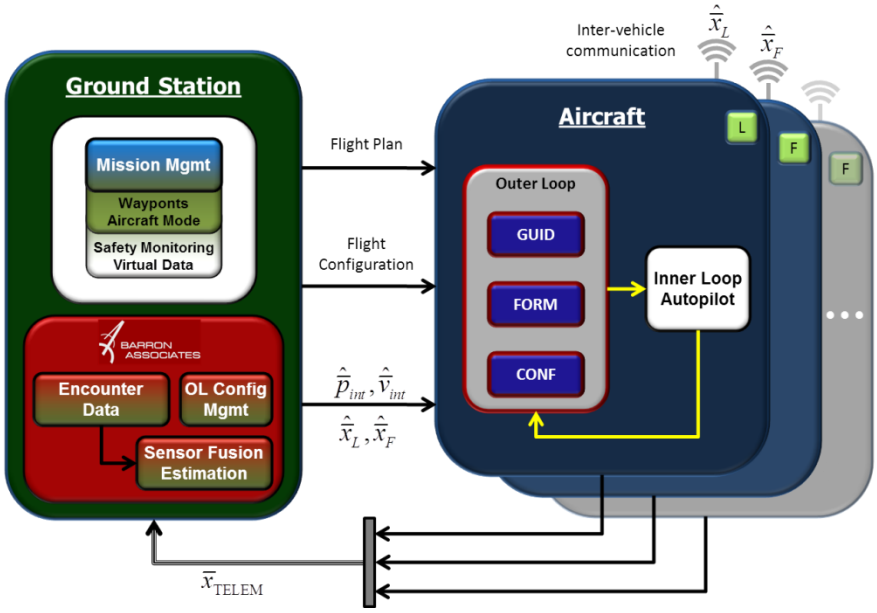

Figure 2. Top-Level MUSAA Architecture

Each vehicle communicates its state to its peers, either directly via intra-fleet communication (air-air) or indirectly via the ground station (air-ground-air). In the figure, $\hat{\bar{x}}_L$ represents the leader's states and $\hat{\bar{x}}_F$ the follower vehicles' states. The simulation architecture developed for MUSAA supports both options by employing variable time delay blocks on telemetry transmitted to fleet vehicles. This architecture also supports transmission of virtual intruder or fleet vehicle data for flight experimentation. The ground station can also generate the flight plan and flight configuration data for vehicle management and test execution. Intruder estimated position, $\hat{\bar{p}}_{int}$, and velocity, $\hat{\bar{v}}_{int}$, and associated uncertainties are also transmitted to the fleet vehicles from the ground station. For actual flight testing, the sensor fusion module would be moved onboard the vehicles. This data is passed to all vehicles in the fleet where the onboard components control the vehicle flight path and respond to potential intruder threats.

B. Design Components

The primary components comprising the MUSAA system include sensor fusion, outer-loop guidance systems, and formation management. Ground station software is developed as needed to support simulation studies and future flight experiments. The onboard MUSAA outer-loop control system is a guidance system which issues commands (bank angle, altitude, and airspeed) to the inner-loop system. Key elements of the MUSAA system are discussed below along with the three outer-loop guidance modes.

Modeling and Simulation Environment: A sophisticated simulation environment has been developed in Matlab/Simulink® throughout the course of this work. The simulation tool allows up to eight simultaneous vehicle simulations within an advanced user interface environment. Additional vehicle simulations can be added, as needed. The level of fidelity of each vehicle is specified by the user. For the current work, medium-high fidelity models are employed. A set of test harnesses have also been developed to comprehensively test and verify the MUSAA system at both the component and integrated levels. Additionally, a version control environment has been integrated with the simulation environment allowing software development by multiple developers simultaneously, with seamless integration of all design iterations.

For this work, we have used a Tier I UAS simulation model. The outer mold line of the vehicle has a standard tailless flying-wing shape and is powered by one central pusher-propeller power plant. It has the following specifications: a maximum takeoff weight of 250 lbf; service ceiling of 20,000 ft; a maximum airspeed of 125 kts; a cruise airspeed of 80 kts; and 24 hours endurance. The tailless flying-wing shape results in poor lateral-directional stability. This model was chosen for its challenging control characteristics.

Waypoint Guidance (Mode Label: GUID): Waypoint guidance provides a representative waypoint following functionality by cycling through a prescribed flight plan of navigation waypoints in latitude, longitude, and altitude format. This guidance mode also serves as the gateway into the other MUSAA outer loop modes and is a candidate safety reversion mode. Note that MUSAA does not depend on this particular waypoint system, but assumes there is some nominal guidance system in place. Some of our testing uses a racetrack pattern as a nominal flight plan which we envision would be desirable for future flight experiments. The majority of testing was performed using this particular waypoint guidance system as the nominal guidance mode. The waypoints are processed in triplets (past, active, and next) to determine the planned trajectory. Waypoint guidance capabilities include flyby waypoints for smoothing the transition between waypoints and fly-over waypoints which force a direct over flight of the waypoint. Waypoints also have altitude, airspeed, and other properties which fully define the flight plan. Rhumbline geometry is used for the trajectory between the waypoints. Example waypoint guidance geometry for a flyby turn is shown in Figure 3.

The path-planning approach leads the vehicle between the waypoints using straight line and circular arc segments. This allows for the separation of ground track geometry from generic guidance trajectory segment calculations and regulation. The guidance approach is designed to generate a bank angle command based on cross-track error and ground track course error between the vehicle and the current position along a line or arc path segment. Altitude tracking is directly controlled by providing desired altitude commands and airspeed tracking is directly controlled by providing desired airspeed commands.

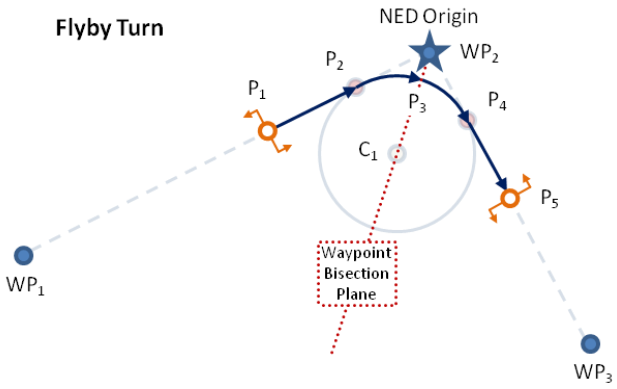

Figure 3. Example Waypoint Guidance Geometry

Formation Flight Guidance (Mode Label: FORM): Since the main objective of this work is to develop SAA algorithms for the multi-UAS application, we are developing a representative leader-follower formation flight control approach. The MUSAA algorithms must be generic enough to work with a variety of formation flight approaches. Formation flight guidance provides a way to create arbitrary formation sizes (number of vehicles in the formation) and geometries for a fleet of aircraft. The formations are designed by specifying slot locations for follower aircraft with respect to the lead aircraft. An additional "level turns" mode is provided which does not change the slot's altitude with changes in the lead aircraft's orientation. The formation flight guidance minimizes the position and velocity error between the follower aircraft and desired formation slot. A bank angle command is generated by analyzing the along- and cross-track error and error rate of the slot with respect to the follower aircraft's body frame. Altitude is directly controlled by providing the altitude of the formation slot. An airspeed command is based on the along-track error and error rate of the slot. An example of the formation flight geometry is shown in Figure 4.

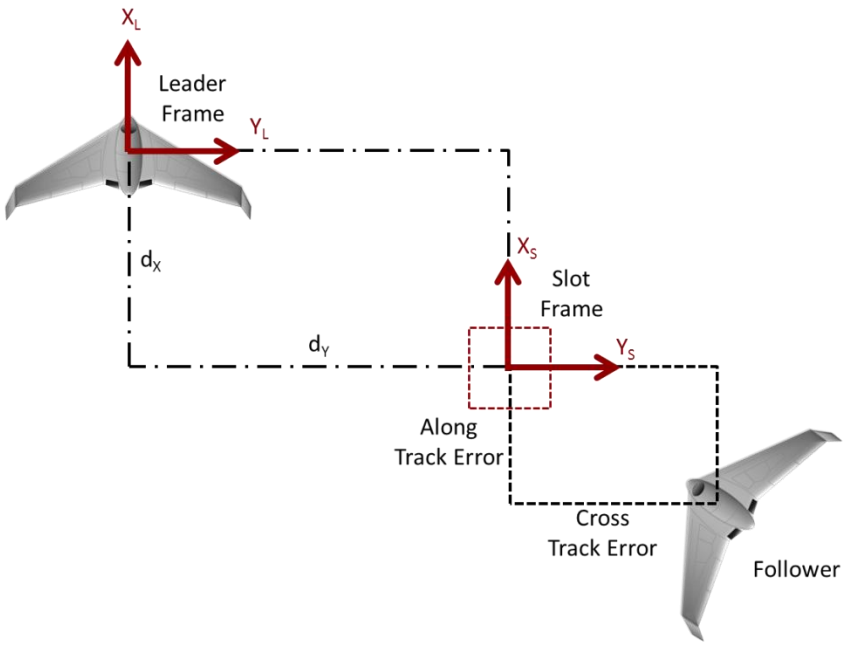

Figure 4. Example Formation Flight Guidance Geometry

Conflict Resolution Guidance (Mode Label: CONF): The conflict resolution algorithms are a primary development area of MUSAA. There are two main components to Conflict Resolution: path planning and path tracking. The planner creates a 3D spatial reference path for the formation to avoid the intruder's projected flight trajectory. The planner resolves the conflict by adhering to a set of constraints defined by community-based rules-of-

the-air and FAA guidelines. Details of the planner are found in Section IV. The path tracker is decoupled into a spatial and a temporal tracker. The path (spatial) tracker is based on an algorithm presented in [4] and modified to meets the needs of the MUSAA system. The path tracker approach follows a spatial path in 3D space and leaves airspeed as a free variable for temporal tracking. The path tracker is shown to be exponentially stable with a specified domain of attraction. [4].

Concurrent with the spatial path-following control law, a temporal coordination component generates airspeed commands in cooperation with other fleet vehicles. This is known as the Airspeed Coordination system. While the path tracking guidance law enforces spatial tracking of a desired path, the Airspeed Coordinator regulates the progression of each vehicle along the path, thereby enabling temporal cooperative behavior of the fleet members and reducing peer-to-peer collision risk. The Airspeed Coordination can also be used to coordinate rendezvous of the formation if the formation splits to resolve a conflict.

Finally, a formation management system is employed in the conflict resolution guidance to determine if the formation should split for a given intruder scenario. It uses a decentralized architecture to leverage the computational power of each vehicle in the formation. The architecture allows for different formation management "strategies" which are a set of rules determining when the formation should split. User-specified strategies enable MUSAA to be deployed in different scenarios such as civilian applications and military hostile theaters, and adapt to changing regulations for integrating UAS in the NAS.

Sensor Data Fusion: The data fusion component of the MUSAA system is used to estimate and predict the motion of the intruder using EO/IR sensors. It provides a baseline capability for sensing the intruder in the absence of more accurate radar or transponder-based information. The use of multiple ownships provides unique opportunities with EO/IR sensors that are not found in the single-ship application. Under this research effort, the team formulated an unscented Kalman filter to fuse the intruder information collected by the formation members. Details are provided in [3].

IV. Path Planning Algorithm Development

A. Review of Path Generation/Planning Approaches

A number of methods have been used to solve path-planning problems including graph-based approaches such as Dijkstra's Algorithm, A*, LRTA*, D*; Monte Carlo or sample-based techniques such as probabilistic roadmaps (PRM) [5] and Rapidly-Exploring Random Trees (RRT) [6]; potential fields, Voronoi diagrams, and optimal control formulation including dynamic programming, and mixed integer linear programming (MILP) [7,8]. These methods can generally be categorized in terms of completeness, computational complexity, optimality, and scalability.

Due to the nonlinear nature of path planning, simplifying assumptions, such as modeling only the kinematics, are often made to create a tractable problem that can be solved online. Such assumptions can have dire consequences for systems where dynamic constraints and control limits define feasible paths a vehicle can traverse. Optimal control formulations that rely on techniques such as variational calculus or Pontryagin's minimum principle become computationally expensive and in some cases intractable for an increased number of vehicles and constraints. Deterministic and complete algorithms require exponential time in the state-space dimension of the dynamic system and polynomial time in the number of obstacles [9]. Therefore, deterministic and complete algorithms are usually implemented for systems with low dynamic dimensions. Even in the single vehicle case the "curse of dimensionality" arises when vehicle dynamics and differential constraints are included (kinodynamic or nonholonomic motion planning) [6]. Some of these methods lack the ability to include constraints on vehicle motion while others simply are too computationally expensive to be practical for online use, especially as the number of vehicles and obstacles increase.

Under a prior research effort led by Northrop Grumman Corp. (NGC), a novel method was developed to triangulate intruder range information using only passive EO/IR sensors and short-duration ownship maneuvers [2]. This approach, termed the Passive Ranging and Collision Avoidance (PaRCA) algorithm, integrated an extended Kalman filter, sensor models, aircraft kinematic and dynamic models, and a modified spherical coordinate system to generate intruder estimates with accuracy sufficient for collision avoidance. Under these prior efforts, the NGC-led team validated the algorithm through a well-developed set of Monte Carlo simulations and a significant number of flight tests, demonstrating promising performance. However, the approach did not address a formation of UASs.

To design a path-planning algorithm that will fit the needs of the current application, a set of requirements must first be established. We propose that an algorithm for cooperative planning suitable for use in a dynamic

environment must (1) execute in real-time, (2) select dynamically feasible actions while avoiding obstacles, (3) utilize the vehicles' full flight envelope, (4) rapidly react and/or replan to account for changing conditions such as mission objectives or obstacle movement, and (5) be able to operate with minimum global information. Furthermore, an architecture encapsulating the path planner should be modular and applicable to different classes of vehicles, e.g. fixed or rotary wing, manned or unmanned.

B. Recurrent Path Planning for a Formation of Aircraft

An optimal conflict resolution path to avoid the intruder has been developed in this effort. The resolution maneuver can be a vertical path, lateral path, or a combination of both, depending on the separation rules that apply. Our solution explicitly satisfies a set of hierarchical constraints based on minimum separation, well clear separation (defined by a hockey puck shape), TCAS/RA requirements, right-of-way (RoW), and other constraints discussed shortly. The path planning algorithm transforms the conflict resolution problem into an iterative static obstacle avoidance problem. The output reference path ensures that all vehicles in the formation avoid the intruder envelope while minimizing deviation of the fleet from the nominal course.

The constraints are based on a set of community-based "rules-of-the-air" and FAA guidelines, illustrated in Figure 5 and summarized below.

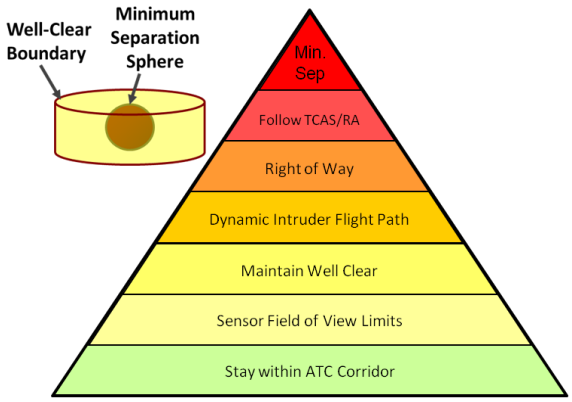

Figure 5. Conflict Resolution "Rules-of-the-Air"

- **Maintain Minimum Separation**: Maintain a minimum lateral and vertical separation distance from other aircraft at all times. The distance is configurable and is set at 820 feet for the purpose of this study. If Minimum Separation cannot be attained, the system shall maximize the separation distance.
- **Observe TCAS Resolution Advisory (RA)**: If the aircraft is unable to meet the specified RA climb / descent rates, the maximum allowable rate will be commanded. The climb / descent rates are the key information utilized in MUSAA, and not the full RA command. It is envisioned that MUSAA could act as an independent system and for this study will detect and react to intruders prior to TCAS engaging either a TA or RA command.
- **FAA Regulations (e.g., 14 C.F.R. § 91.113)**: 14 C.F.R. §91.113 addresses converging, head-on, and overtaking scenarios and establishes the aircraft with the right-of-way in these conditions.
- **Dynamic Intruder Flight Path**: Respond to, and update, the resolution path based on the intruder's dynamic behavior.
- **Maintain Well Clear**: The path should maintain a minimum lateral and vertical well clear separation distance from other aircraft. The distances are configurable and selected as 2460 feet laterally and 820 feet vertically for the purpose of this study. If Well Clear cannot be attained, the system shall maximize the separation distance.
- **Keep Sensor Field of Regard**: Given a set of sensors, the UAS should keep the intruder aircraft in its field of view while the separation rate is negative and to the extent possible, while the intruder aircraft is deemed a potential threat. Numerical values for the field of view in azimuth and elevation range and detection distance are dependent on the sensor suite; however, nominal values are selected as:
  - Azimuth FOV: [-110^o, 110^o] measured from the nose of the aircraft

- Elevation FOV: [-15°, 15°] measured from the flight path angle of the aircraft.

- **Maintain ATC Corridor**: Any maneuver to resolve a conflict should minimize the deviation from the nominal/desired path.

A key feature of our approach is a fully analytical formulation that rapidly generates a complete reference path. We take advantage of this to call the path planning algorithm at a specified update rate to drive the guidance loop and, therefore, reshape the path in real-time to react to intruder motion. The path planner uses a "reactive separation" approach which dynamically adjusts the avoidance path to satisfy the required separation from the intruder. The algorithm propagates an internal point-mass model for the ownship and intruder and stores the ownship's position data at each propagation step. Once the internal propagation is completed, the ownship position data is provided as a reference path to track in real-time.

Internal Propagation: The conflict resolution algorithm propagates an internal model of the intruder's position (assuming constant airspeed and flight path) and simultaneously propagates an internal model of the ownship position, redirecting the ownship path in order to avoid the intruder's projected path.

At each internal propagation step the algorithm solves a static optimization problem to select the heading and/or climb rate command which will cause the ownship to avoid the intruder by the well clear distance at the time of closest point of approach (CPA, τ_{CPA}). If the well clear distance cannot be achieved, the algorithm maximizes the distance at the time of CPA $(D_{\min})$. For the case of a lateral maneuver, an optimal heading command is calculated and passed to the point-mass dynamic model and the ownship position is propagated forward by one time-step while attempting to track the heading command. Simultaneously, the intruder states are propagated forward one time-step.

The appropriate ownship heading command to achieve the maximum $D_{\min}$ is determined using a very rapid optimization based on analytical models of the ownship and intruder kinematics. The time to minimum separation and $D_{\min}$ are nonlinear functions of the ownship and intruder positions $(\vec{r}_o, \vec{r}_i)$ and velocities $(\dot{\vec{r}}_o, \dot{\vec{r}}_i)$. Calculating τ_{CPA} is accomplished with the concept of a cylinder norm described in [10]. The cylinder norm is defined for a vector $\mathbf{w} \equiv (w_i, w_j, w_k)$ as

$$\|\mathbf{w}\|_{cyl} = \max\left(\frac{|w_k|}{vertWC}, \frac{\|\mathbf{w}_{(i,j)}\|}{latWC}\right) \tag{1}$$

where $\mathbf{w}_{(i,j)} \equiv (w_i, w_j)$ is the projection of $\mathbf{w}$ on the horizontal plane, vertWC is the defined vertical well clear distance, and latWC is the defined lateral well clear distance. We also introduce bounds on the time variable to indicate how far in the future we will look for a potential collision; $\tau_{CPA} \in [t_{min}, t_{max}]$. Similar to the Euclidean norm, the time to CPA is found by minimizing the square of a norm, in this case the cylindrical norm, $\|\Delta\mathbf{p} + t\Delta\mathbf{V}\|_{cyl}^2$. Expanding this statement yields,

$$\begin{aligned} \|\Delta\mathbf{p} + t\Delta\mathbf{V}\|_{cyl}^2 &= \max\left(\frac{(\Delta p_z + t\Delta V_z)^2}{vertWC^2}, \frac{\|\Delta\mathbf{p}_{(x,y)} + t\Delta\mathbf{V}_{(x,y)}\|^2}{latWC^2}\right) \\ \Delta\mathbf{p} &= \mathbf{r}_i - \mathbf{r}_o \\ \Delta\mathbf{V} &= \dot{\mathbf{r}}_i - \dot{\mathbf{r}}_o \end{aligned} \tag{2}$$

Equation (2) involves two quadratic polynomials and a maximization function. The minimum of this function is found by satisfying one of the following conditions [10],

- $\tau = t_{min}$,
- $\tau = t_{max}$ and $t_{max} \neq \infty$,
- $\tau = \dfrac{-\Delta p_z}{\Delta V_z}$ or $\dfrac{-\Delta\mathbf{p}_{(x,y)} \cdot \Delta\mathbf{V}_{(x,y)}}{\|\Delta\mathbf{V}_{(x,y)}\|^2}$,

- τ is the solution to the system obtained by setting the two quadratic polynomials in Eq. (2) equal to each other,

$$\frac{\left(\Delta p_z + t\Delta V_z\right)^2}{vertWC^2} = \frac{\| \Delta \mathbf{p}_{(x,y)} + t\Delta \mathbf{V}_{(x,y)} \|^2}{latWC^2} \Rightarrow At^2 + Bt + C = 0 \tag{3}$$

where,

$$A = \frac{\Delta V_z^2}{vertWC^2} - \frac{\| \mathbf{v}_{(x,y)} \|^2}{latWC^2},\ B = \frac{2\Delta p_z \Delta V_z}{vertWC^2} - \frac{2\Delta \mathbf{p}_{(x,y)} \cdot \Delta \mathbf{V}_{(x,y)}}{latWC^2},\ C = \frac{\Delta p_z^2}{vertWC^2} - \frac{\| \Delta \mathbf{p}_{(x,y)} \|^2}{latWC^2}$$

The solution of Eq. (3) has two unique real solutions if the discriminant is greater than zero and two equal solutions if the discriminant is equal to zero. Complex solutions exist if the discriminant is less than zero and are not a valid solution in this case. This then results in six possible solutions for τ_{CPA} which are each evaluated with the cylindrical norm to determine the time of closest approach.

The minimum separation distance is described as

$$D_{\min} = \left\| \Delta \mathbf{p} + \Delta \mathbf{V}\left(\chi_c\right) \tau_{CPA}\left(\chi_c\right) \right\| = f\left(\chi_c\right) \tag{4}$$

where χ_c is the ownship heading direction. A simple secant-search algorithm is used to determine the heading reference that minimizes the error between the minimum separation distance and the desired well clear separation distance,

$$\min_{\chi_c \in \left[\chi_i - \pi/2, \chi_i + \pi/2\right]} \left(D_{\min}(\chi_c) - R_{well\ clear} \right) \quad \Rightarrow \quad \chi_c^* \tag{5}$$

Once the optimal heading reference is solved, the bank angle command to achieve this heading is given by,

$$\mu_c = k_\mu \left. v_0 \right/ g \left(\chi_c^* - \chi_i \right) \tag{6}$$

which can be limited at an operator prescribed level. The ownship vehicle states are then propagated forward to the next time step using the bank angle command. This procedure repeats until the computed separation rate becomes positive and the threat has passed, or is "cleared". Recovery logic then commands a heading angle to steer the ownship model back to the original course, driving the cross track error to zero. Similar logic is used for vertical separation to determine a vertical flight path command, γ_c, to achieve a well clear separation.

Figure 6 illustrates the internal ownship path propagation performed during a single call to the conflict resolution algorithm. The internal propagation time is denoted τ and the ownship desired heading and flight path angle are determined from the optimization algorithm at each time-step, τ_i. The complete internally propagated path (green path in Figure 6) is provided as the reference path for the ownship to track in real time.

The path planner accounts for a dynamically maneuvering intruder by re-planning, or updating, the resolution path at a user-defined rate. Figure 7 shows two cases where the intruder turns into and away from the ownship. In each case the ownship is initialized at the origin. The blue and black marks show the actual path of the ownship and intruder, respectively. The red lines show the planned resolution trajectories and the green lines represent the propagated intruder path at each update time. When the intruder turns into the ownship, the path planner progressively updates the path with greater deviation from the nominal path to achieve well clear, while minimizing deviation from the nominal path. As expected, this trend is reversed when the intruder turns away from the ownship.

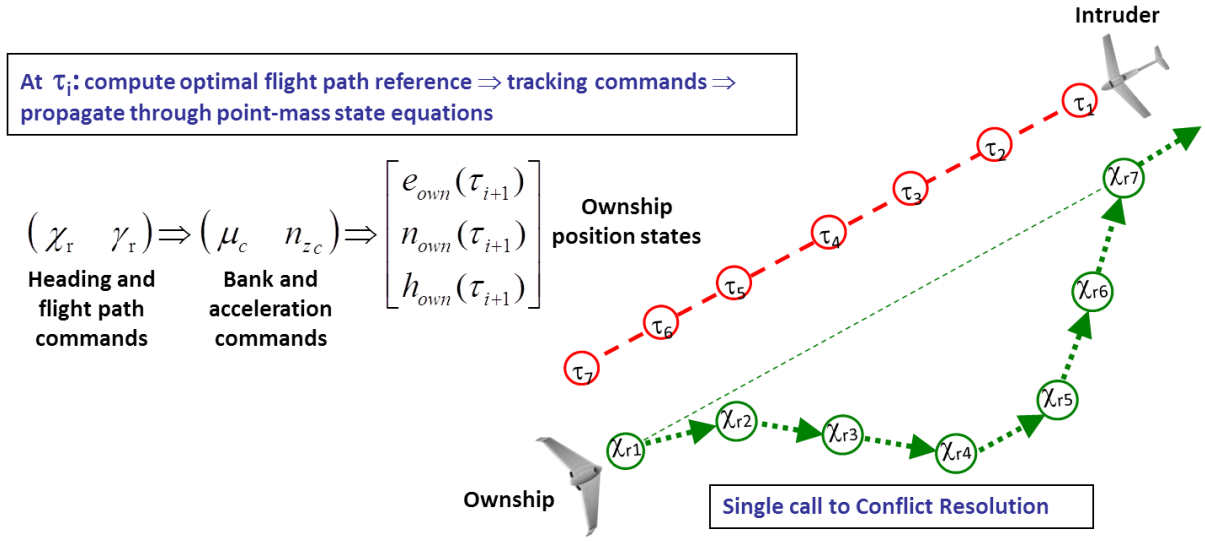

Figure 6. Ownship Path Propagation for Conflict Resolution Algorithm

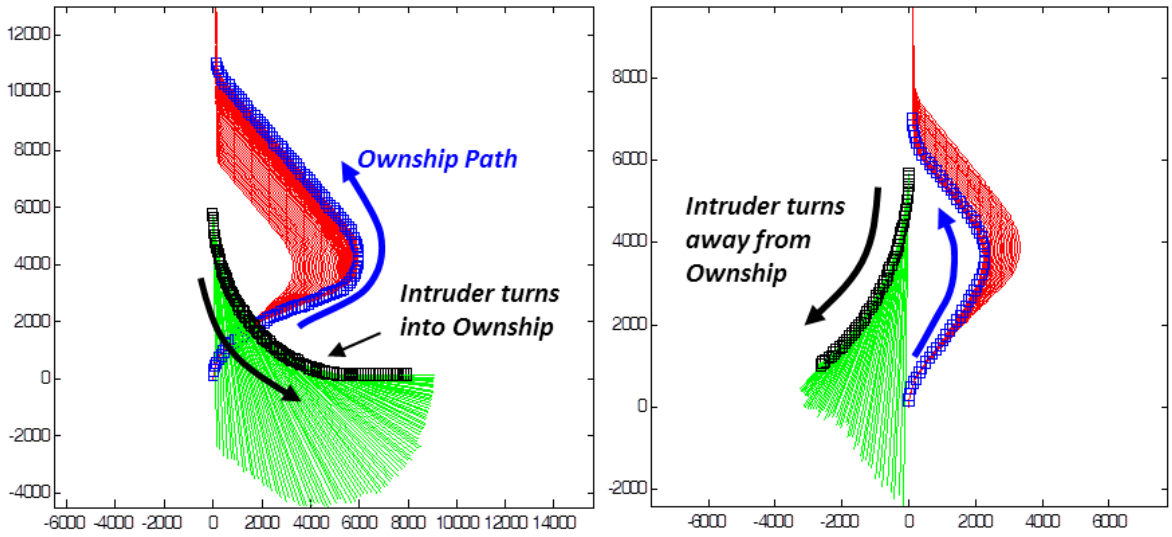

Figure 7. Resolution Path Reacts to Dynamic Intruder

Path Parameterization with Bspline Curves: The reference path resulting from the internal point-mass propagation is a set of 3D data points, too large for communication and path de-confliction if the formation needs to split. An efficient and accurate means is required to represent the propagated points which constitute the conflict resolution path. This allows the path to be transmitted to formation members with minimum required bandwidth. The formation members can then reconstruct the path onboard to ensure path de-confliction. We model the path with a Bspline curve [11]. Bspline curves are segmented into so-called knot spans, or intervals, by a knot vector $U = \{u_i\}_{i=0}^{m} \in \Re^{m+1}$, with the ith knot span defined on the interval $[u_i, u_{i+1})$. Each interval contains a single Bspline basis function, a polynomial of degree p, $N_{i,p}(u)$, and control point, $\bar{P}_i$. The Bspline curve, $\mathcal{C}(u)$, is the weighted sum of the basis functions and control points:

$$\mathcal{C}(u) = \sum_{i=0}^{n} N_{i,p}(u)\,\bar{P}_i \tag{7}$$

The Bspline curve has order "n" with basis functions of degree p. The order and degree are related with the number of control points, viz., $m = n + p + 1$. Bspline basis functions can be defined by the following recurrence relation:

$$N_{i,0}(u) = \begin{cases} 1; \text{if } u_i \le u \le u_{i+1} \\ 0; \text{otherwise} \end{cases} \tag{8}$$

and for higher order functions,

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u) \tag{9}$$

Figure 8 graphically portrays Bspline basis function of zero, first, and second order. For a given spline order, the domain of interest is "covered" by shifting versions of the spline.

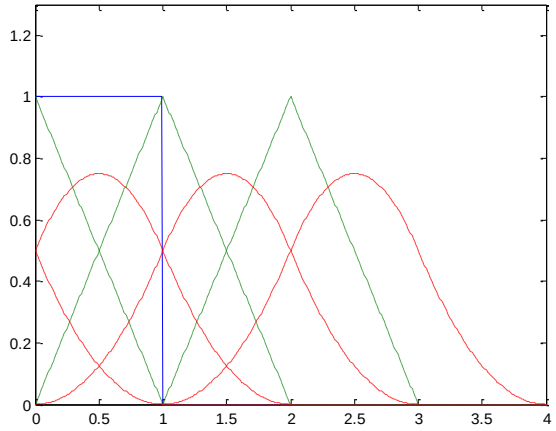

Figure 8. B-Spline Basis Functions of First (-), Second (-.) and Third (--) Order

Bspline curves have many useful mathematical properties relevant to the MUSAA program. A few of the more salient properties are listed below:

- **Local Support and Modification**: A Bspline basis function, $N_{i,p}(u)$, of degree p is non-zero on p+1 knot intervals $[u_i, u_{i+p+1}]$. If $u \notin [u_i, u_{i+p+1}]$ then $N_{i,p}(u) = 0$. This means Bsplines have local support, or are active over a local portion of the overall domain. If a modification is made to a control point $\bar{P}_i$ then the effect on the Bspline curve is contained to the interval $[u_i, u_{i+p+1}]$.
- **Strong Convex Hull Property**: The Bspline curve $\mathcal{C}(u)$ is contained within the convex hull of a control polygon defined by its control points $\{\bar{P}_i\}_0^n$. Also, if $u \in [u_i, u_{i+p+1}]$ then $\mathcal{C}(u)$ lies within a smaller convex hull defined by the p+1 control points $\{\bar{P}_k\}_{i-p}^{i}$. This property is useful for path de-confliction and evaluating points along the curve and its derivatives. Path de-confliction can be assessed by inspection of the control polygon; evaluating points along the path may not be necessary.
- **End point Conditions**: Bspline curves that have the first and last knots with a multiplicity of p+1, have the desirable properties that the curve interpolates the first and last control points, and is tangent to the control polygon at these points.

Least-squares Path Approximation: A least squares technique is used to approximate the conflict resolution path. The propagated points are stored in a $Nx3$ matrix and a Bspline curve is approximated to match the data points. The result is a matrix of control points, $P \in \Re^{n+1x3}$. Figure 9 shows a representative comparison of the propagated points and Bspline approximation. In this case the curves nearly overlay each other.

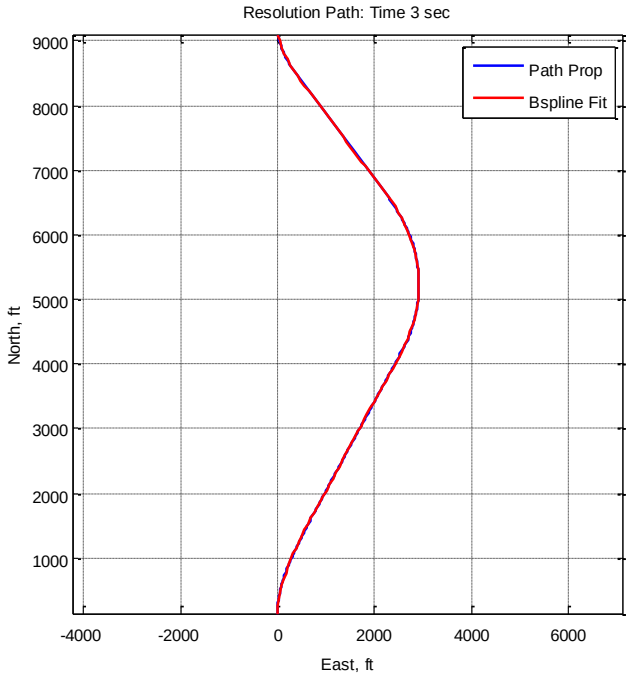

Figure 9. Comparison of Propagated Points and Bspline Approximation

V. Example Experimental Results

A. Single Intruder Scenario

A number of intruder scenarios were created and used to evaluate the MUSAA system. These scenarios were created by varying the intruder's initial position, heading, and climb rate relative to the lead aircraft of the UAS formation. The intruder maintained its initial flight plan and speed profile in some cases, while in others the airspeed, bank angle, and altitude were varied. Some of the intruder scenarios created instances where the well clear distance could not be satisfied due to the initial conditions and dynamic limitations of the fleet. These cases allowed evaluation of MUSAA under more demanding resolution scenarios. We present one such example case in which the intruder is detected late, causing a breach in the well clear distance. For this case the intruder has the initial conditions summarized in Table 1. All initial conditions are relative to the leader aircraft.

Table 1. Intruder Initial Conditions Relative to Lead Aircraft (Case 1)

dX (ft)	dY (ft)	dZ (ft)	dPsi (deg)	hdot (fps)	gamma (deg)
4500	0	0	-180	-40	-15.2575

The resolution path for this case is shown in Figure 10 along with ancillary information. The following information is shown in the figure:

- Initial leader and intruder positions,
- Nominal leader and follower paths,
- Resolution path for the leader,
- Point at which the intruder is deemed "clear" and it is safe to return to the nominal path; this is indicated with a red square on the leader and intruder's paths,
- Adherence to safety distances: this is overlaid on the resolution path with color coded markings: green indicates lateral or vertical well clear distances are satisfied, yellow indicates lateral or vertical minimum separation

distances are satisfied, and red indicates a violation of the minimum safety distances. This same information is also overlaid on the intruder's position data.

- Maintaining Sensor FOV: a track indicating the intruder is in the sensor FOV is plotted above the resolution path and the intruder's path. A green marker indicates the intruder is within the FOV, yellow indicates the intruder is within the lateral or vertical range, but not both, and red indicates the intruder is outside the sensor FOV.
- Initial Time to CPA: The time to CPA based on initial conditions is displayed at the leader's initial position.
- Maximum Lateral Deviation: displayed at the red square, indicating the intruder has been cleared.

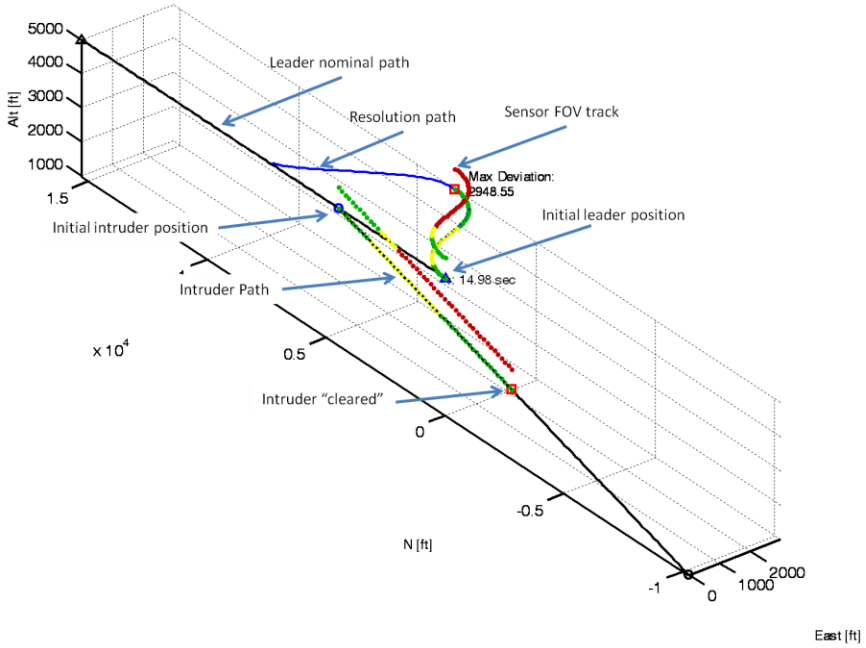

Figure 10. Resolution Path for Case 1

If the safety volume was breached a plot similar to Figure 11 shows minimum predicted separation distance. A figure showing the lateral and vertical separation distances along the length of the resolution path is also generated as shown in Figure 12. The vertical line in this figure indicates when the intruder has been "cleared". For this case the intruder was detected with a time to CPA of 14.98 seconds which is an extremely late detection time. Consequently, lateral well clear was not maintained but minimum separation was maintained. This metric is dependent on the maneuvering capabilities of the vehicle which is an input to the MUSAA software.

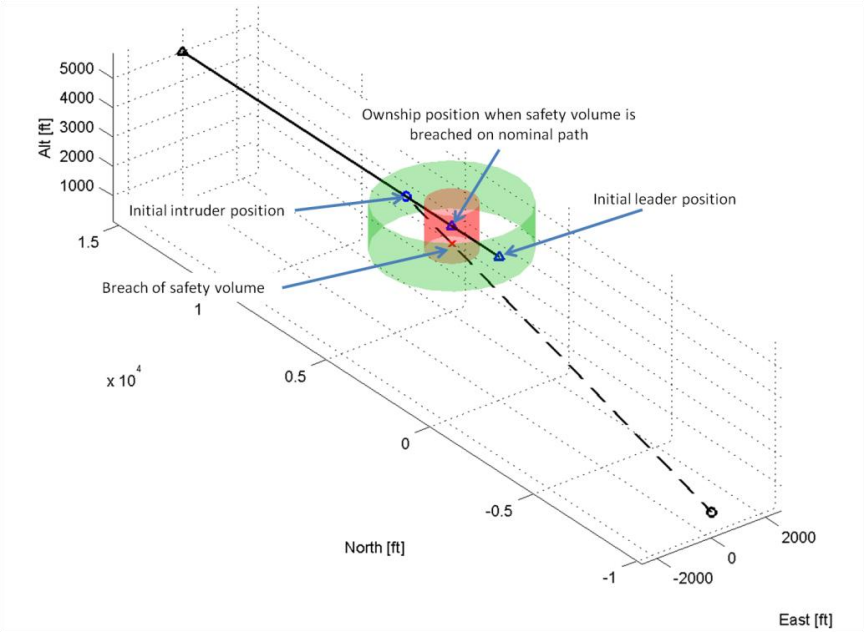

Figure 11. Safety Volume for Case 1

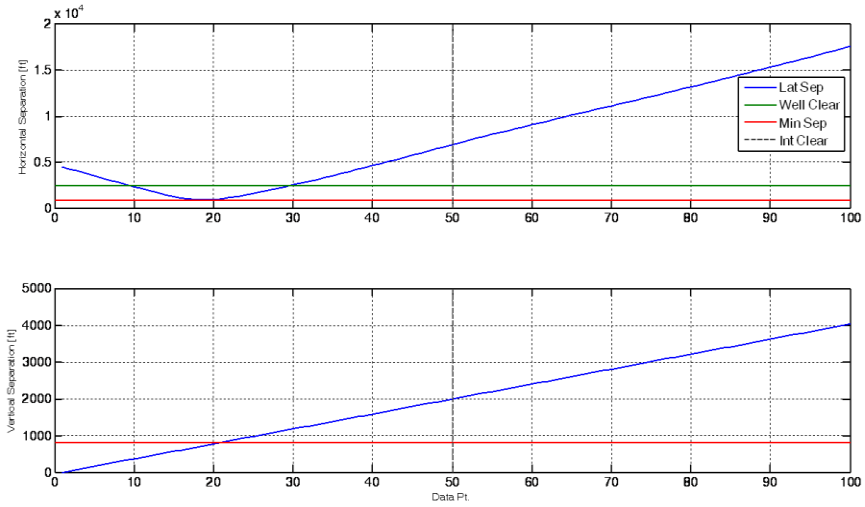

Figure 12. Lateral and Vertical Separation Distances, Case 1

B. Sequential Intruder Scenarios

Sequential intruders indicate that MUSAA currently plans a resolution path for one intruder at a time, hence, the phrase "multiple intruders" is not used. MUSAA currently assumes the first, or initial, intruder has been "cleared" prior to the appearance of the next intruder. There are two general sequential intruder scenarios. The first is termed "distant sequential" where the formation completes a conflict resolution maneuver prior to the appearance of the second intruder. This scenario can be decomposed as:

- Distant Sequential Form – Form: The formation is maintained for the first intruder; the resolution maneuver is completed; the formation is maintained for the second intruder.
- Distant Sequential Split – Form: The formation is split for the first intruder; the sub-formations complete the resolution maneuvers and the formation is regrouped; the formation is maintained for the second intruder.

The second general scenario is termed "close sequential" where the formation does not complete a resolution maneuver prior to the second intruder appearing. The different cases for this scenario include:

- Close Sequential Form – Form: The formation is maintained for the first intruder; the formation is "clear" of the first intruder and then the second intruder appears; the formation is maintained for the second intruder.
- Close Sequential Split – Form: The formation is split for the first intruder; the sub-formations "clear" the first intruder, but still in CONF mode, and then the second intruder appears; the sub-formations are maintained for the second intruder; the sub-formations regroup after resolving the second conflict.
- Close Sequential Form – Split: The formation is maintained for the first intruder; the formation is "clear" of the first intruder, but still in CONF mode, and then the second intruder appears; the formation is split for the second intruder; the sub-formations regroup after resolving the second conflict.
- Close Sequential Split – Split: The formation is split for the first intruder; the sub-formations are "clear" of the first intruder and then the second intruder appears; another split occurs for the second intruder; the sub-formations regroup after resolving the second conflict.

The following sections summarize results for the cases where the formation is maintained: Distant Sequential Form – Form and Close Sequential Form – Form. A future article will discuss the case where the formation splits.

Distant Sequential Form – Form: The ground tracks of the formation and intruder vehicles for this scenario are shown in Figure 13. The formation is headed due north and the intruders are headed due south. The first intruder is offset by 10,000 feet from the formation. There are three ownships in the formation. The initial conflict is resolved with a lateral maneuver to the right. The formation completes the resolution maneuver and returns to the nominal path. Subsequently, a second intruder is detected and a resolution path to the left is generated, owing to the offset position of the intruder.

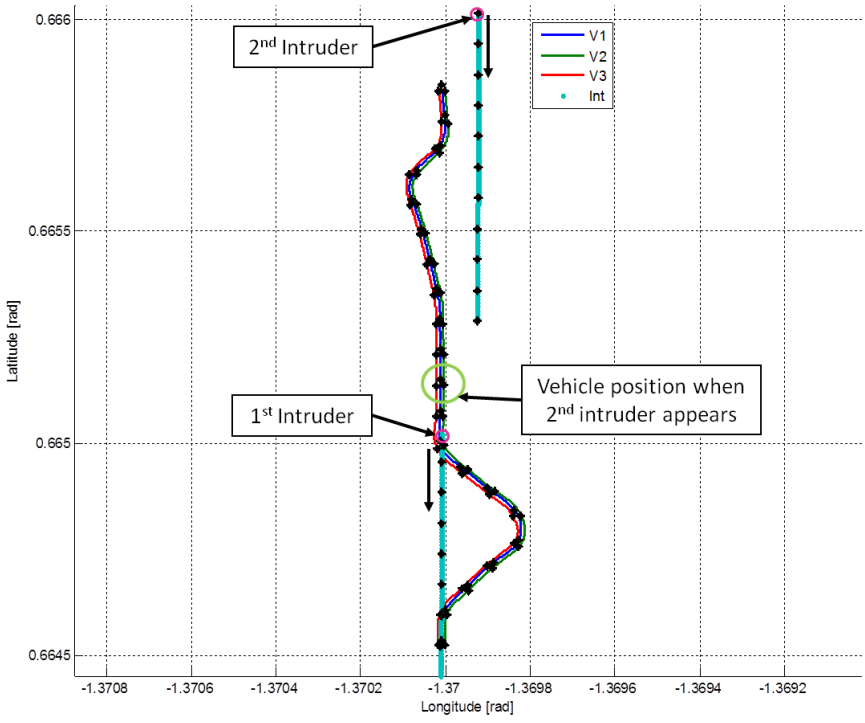

Figure 13. Distant Sequential Form-Form Ground Track

Close Sequential Form – Form: The ground tracks of the formation and intruder vehicles for this scenario are shown in Figure 14. The formation consists of three ownships and is headed due north. The intruders are headed due south. The first intruder is offset by 10,000 feet from the formation. The resolution path is a lateral maneuver to the right. When the second intruder appears the path planner adapts the path to achieve well clear for the formation.

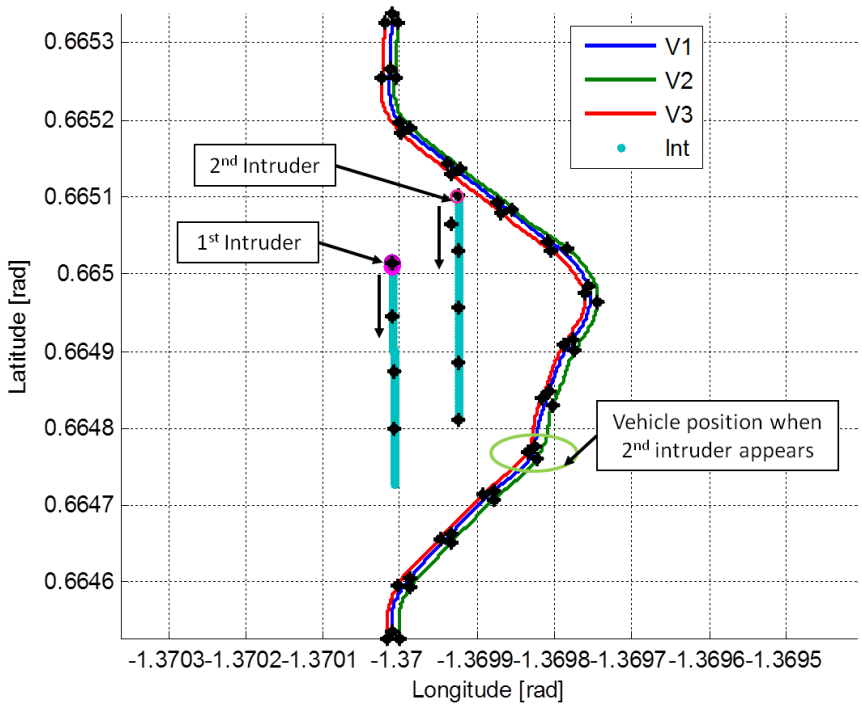

Figure 14.Close Sequential Form – Form Ground Track

VI. Evaluation of MUSAA Algorithms on Embedded Hardware

Initial embedded testing of the MUSAA algorithms on representative flight control hardware has been conducted. The objective of these tests was to verify the ability to run MUSAA technology on a processor with limited computational ability in preparation for flight experiments. The authors evaluated the MUSAA algorithms on a Gumstix Overo FireSTORM COM. Initial tests show the MUSAA software runs in the desired frame rate (10 Hz), and does not stress the memory and processor restrictions of the Overo COM.

A. Code Generation

The MUSAA algorithms were ported to the Gumstix board using a set of code generation tools provided by The Mathworks and tools generated in-house. The Mathworks offers a number of add-on products to generate embedded software and code from Matlab / Simulink development software. These products were utilized to generate code that can be built on an embedded Linux target, although the software is developed on a Windows host. A number of third party tools were also required. BAI modified some of the provided Mathworks tools to make the process more transparent and better suite project needs. Figure 15 depicts a high-level overview of the process. On the Windows host computer, the MUSAA algorithms are converted to C code using Mathworks tools. The result is a set of source code files which is packaged and transmitted to the Overo COM. Once on the Overo COM, the source files are compiled using the tool-chain and compiler of the COM. We can then run the program, collect data, and transmit this data back to the host computer for analysis. A simple test program was created to verify this process. A Python script was written to log CPU usage, memory requirements, and timing information of the process.

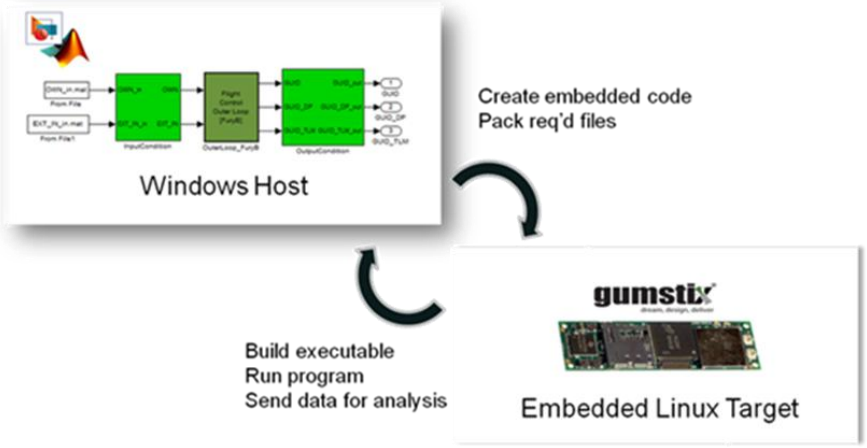

Figure 15. Generate Embedded Code from Development Software

B. Test Results

The test scenario included an intruder and ownship in a "head-on" collision, initially separated by 10,000 feet. The ownship operates in the Waypoint and Conflict Resolution guidance modes. Although this initial test does not provide complete code coverage, the Conflict Resolution guidance mode is exercised and is the most computationally complex portion of MUSAA. Figure 16 illustrates results of the intruder scenario and resolution path. The latitude and longitudinal position of the ownship is shown by the red line, intruder position by the black line and waypoints by blue circles.

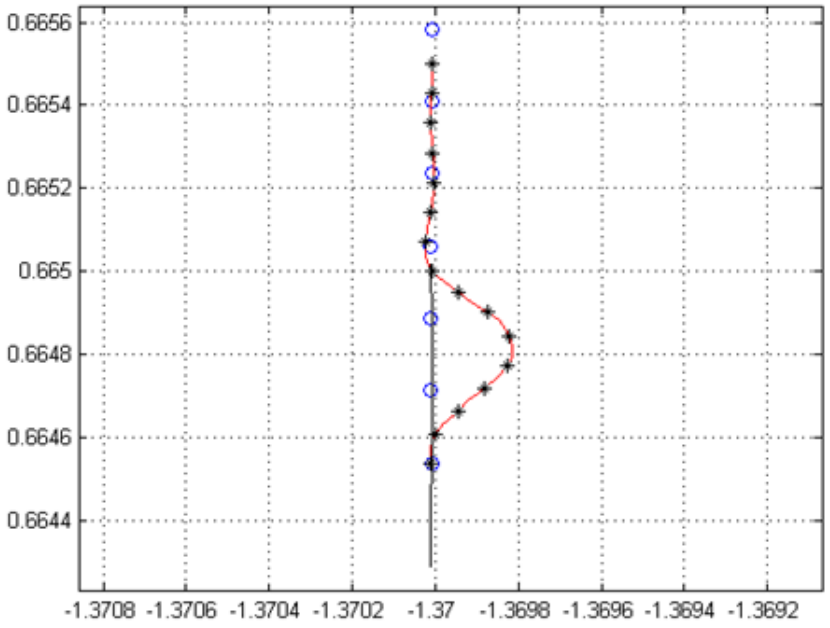

Figure 16. Intruder Scenario for Testing Embedded Code

Two timing metrics were collected with the Overo COM. The first, Call Time, is the time between calls to the MUSAA algorithms. The expected call time is 0.1 seconds. The first subplot in Figure 17 illustrates the Call Time occurs extremely close to the desired rate. The second timing metric is the Execution Time and represents the amount of time required to run the MUSAA algorithms each time it is called. This metric is presented in the lower subplot of Figure 17. Again, the timing is well within the 0.1 second requirement. A number of spikes are observed prior to the 400th data point. These spikes occur at a 1Hz rate and correspond to calls to the path planner, when MUSAA updates the desired path. The guidance laws operate at 10Hz. Shortly after the 800th data point, MUSAA

exits Conflict Resolution guidance mode and enters waypoint guidance. During this run, the approximate maximum CPU load related to MUSAA was 58% and occurred during calls to the CONF path planner. When the path planner was not being executed MUSAA required approximately 1-2% of the processing power.

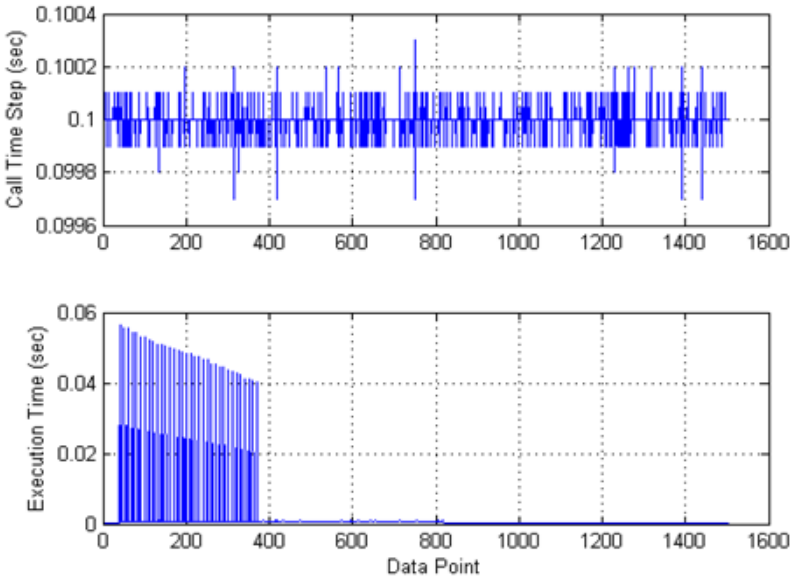

Figure 17. Timing Metrics on Gumstix COM

VII. Conclusions and Recommendations

A sense and avoid (SAA) system has been developed for multi-UAS platforms in formation flight. The "sense problem" is addressed by using intra-fleet communication and a sensor fusion algorithm. With this, the formation aircraft effectively form a stereo-optic solution by sharing their sensory information. This solution is then used in the core of the SAA system, which addresses the "avoid problem." This is a conflict resolution algorithm that generates a resolution path to satisfy a hierarchical set of safety constraints including a pre-defined "well clear" volume. If the well clear criterion cannot be achieved, the conflict resolution algorithm maximizes the separation distance. The conflict resolution algorithm can generate a lateral avoidance maneuver, a vertical avoidance maneuver, or both, depending on the scenario and a user-defined resolution "strategy." The airspeed of the formation can also be adjusted to aid in conflict resolution.

The developed SAA algorithm is shown to have low computational overhead. The required intra-fleet communication bandwidth is also minimized through the use of Bsplines to model the commanded resolution path. The conflict resolution algorithm to continuously update its solution, adapting in real time to changes in the intruder's path. Therefore, the overall solution does not require the assumption that the intruder has constant heading and constant airspeed, which is assumed in prior SAA development work. Further, by adapting to the intruder's maneuvering, the SAA solution is able to minimize the departure from the nominal path, resulting in a less conservative approach. This has the advantage of minimizing air traffic management considerations, fuel expenditure, and overall effect on the nominal mission.

Component and integrated testing of the sensor fusion, conflict resolution and other guidance modes have been accomplished and show promising results. The SAA system is shown to be robust to a large space of initial conditions and scenarios. To date, the SAA system has been analyzed with homogeneous and heterogeneous fleets of Tier I UAS platforms. Investigation of the SAA design with Tier II or III platforms is planned. Robustness to modeling errors, winds, and other environmental conditions will also be investigated. It is recommended to further mature the system to handle multiple, simultaneous intrusions. Flight experimentation of the SAA system is currently in progress using experimental platforms. This will help to lay the groundwork for future flight testing of the SAA system in UAS platforms of interest.

Acknowledgements

This work was funded by the Air Force Research Laboratory at Wright-Patterson Air Force Base under contract no. FA8650-10-C-3010. We would like to thank Stephanie Simon, Jane Thompson, Vince Raska, and their team for their roles in overseeing this work. Their support, feedback and interest is greatly appreciated. We also wish to express our gratitude to Naira Hovakimyan and Vladimir Dobrokhodov for their support and expertise during the course of the project.

References

1. Shakernia, O., Chen, W., Graham, S., Zvanya, J., White, A., Weingarten, N., Raska, V., "Sense and Avoid (SAA) Flight Test and Lessons Learned," AIAA Infotech@Aerospace Conference and Exhibit, Rohnert Park, CA, May 2007, AIAA 2007-3003.

2. Shakernia, O., Chen, W., Raska, V., "Passive Ranging for UAV Sense and Avoid Applications," AIAA Infotech@Aerospace Conference and Exhibit, Arlington, VA, September 2005, AIAA 2005-7179.

3. Lichter, Matthew D., Cooper, Jared, K., Schierman, John D., "Sensor Fusion within Formation of Unmanned Aircraft, with Applications to Sense-and-Avoid Systems", 2014 American Control Conference, June 4-6 2014, Portland, OR (*to appear*)

4. Kaminer, Isaac, Pascoal, Antonio, Xargay, Enric, Hovakimyan, Naira, Cao, Chengyu, and Dobrokhodov, Vladimir, "Path Following for Unmanned Aerial Vehicles Using L1 Adaptive Augmentation of Commercial Autopilots", *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 2, March-April 2010.

5. L. E. Kavraki, S. P., L. J. C., and M. H. Overmars, "Probabilistic Roadmaps for Path Planning in High-Dimensional Configuration Spaces," in *IEEE Transactions on Robotics and Automation*.

6. S. M. LaValle and J. J. Kuffner, "Randomized Kinodynamic Planning," in IEEE International Conference on Robotics and Automation.

7. T. Schouwenaars, B. DeMoor, E. Feron, and J. How, "Mixed Integer Programming for Multi-Vehicle Path Planning," in Proceedings of the European Control Conference, Sep 2001.

8. N. Richards and R. Bird, "UAV 2D and 3D Path Planning for Sensor-on-Target Maneuvers and Obstacle Avoidance," Barron Associates Technical Report 294, prepared for Northrop Grumman Software Enabled Control.

9. E. Frazzoli, M. A. Dahleh, and E. Feron, "Real-Time Motion Planning for Agile Autonomous Vehicles," *Journal of Guidance, Control and Dynamics*, vol. 25, pp. 116-129, January-February 2002.

10. César A. Munoz and Anthony J. Narkawicz, "Time of Closest Approach in Three-Dimensional Airspace," Technical Report, National Aeronautics and Space Administration, October 2010, NASA/TM2010216857.

11. Sederberg, Thomas W., *Computer Aided Geometric Design*, January 2011.