IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received November 22, 2019, accepted December 15, 2019, date of publication January 15, 2020, date of current version January 27, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2966763

On Using Composability Tools for Reliability Analysis of Unmanned Multi-Aircraft Systems: A Case Study

DEVAPRAKASH MUNIRAJ[1], DANY ABOU JAOUDE[2], AND MAZEN FARHOOD[1]
[1]Kevin T. Crofton Department of Aerospace and Ocean Engineering, Virginia Tech, Blacksburg, VA 24061, USA
[2]Department of Mechanical Engineering, American University of Beirut, Beirut 1107-2020, Lebanon

Corresponding author: Mazen Farhood (farhood@vt.edu)

This work was supported in part by the National Science Foundation (NSF) under Grant CMMI-1351640 and Grant CNS-1801611, and in part by the Center for Unmanned Aircraft Systems (C-UAS), an NSF sponsored industry/university cooperative research center (I/UCRC), under NSF Grant IIP-1539975 and Grant CNS-1650465, along with significant contributions from C-UAS industry members.

ABSTRACT This paper presents a case study that demonstrates how tools from compositional verification can be used to design and analyze complex multi-agent systems operating in dynamic and uncertain environments. The case study concerns the design of an unmanned multi-aircraft system tasked to compromise an aerial encroacher by deploying countermeasures. The constituent agents, termed defenders, are fixed-wing unmanned aircraft. To successfully compromise the encroacher, at least one defender must be within a prespecified distance from the encroacher for a certain period, and the defenders must avoid collision among themselves and with the encroacher. Verifying this global property using monolithic (system-level) verification techniques is a challenging task due to the complexity of the components (defenders) and the interactions among them. To overcome these challenges, the components are designed to have a modular architecture, thereby enabling the use of component-based reasoning to simplify the task of verifying the global system property. Results from Euclidean geometry and formal methods are used to prove most component properties. For properties where analytical tools are overly conservative, focused Monte Carlo simulations are carried out. Restricting the use of simulations (or testing) to local verification of partial component properties leads to increasing the reliability of the system.

INDEX TERMS Compositional reasoning, formal verification, system analysis and design, temporal logic of actions, unmanned aerial vehicles.

I. INTRODUCTION

A fundamental challenge in deploying a multi-agent cyber-physical system (CPS), such as a network of unmanned aircraft systems (UAS), in a safety-critical application is expressed by the question: *How can one assess and ensure the reliability of such a complex system?* This question becomes all the more challenging when considering systems that are required to execute missions in an uncertain environment [1] while being subject to threats that could be internal (e.g., a component failure) as well as external (e.g., environmental hazards). Given the importance of establishing trust in multi-agent CPS before their deployment in safety-critical applications, certification agencies require the designers of such systems to provide an assurance case, which is a structured argument backed by evidence that is intended to justify that the designed system is safe and secure. The prevailing approach to building an assurance case involves conforming to standards and guidelines, such as DO-178C [2] and DO-254 [3], that specify objectives for design and certification of safety-critical flight software and hardware, and conducting extensive simulations and tests on the designed system to generate the necessary evidence. The analyst is faced with the challenge of evaluating the system for all possible inputs and operational scenarios, which can be prohibitive for a multi-agent CPS composed of highly-interacting components. Even if one adopts techniques from reliability engineering and statistics to reduce the number of simulations and tests, the stringent failure rate requirements imposed by standards such as [2] and [3] would still necessitate considering a large number of test cases. Moreover, any simulation or testing method would only be able to test a limited number

The associate editor coordinating the review of this manuscript and approving it for publication was Jiankang Zhang.

of operational scenarios [4]. Given the time and resource constraints, it is therefore desirable to employ techniques that would restrict the use of simulations and tests to specific subsystems.

It has been recognized that no one method is adequate for verification and validation of complex CPS, and there is a need to employ techniques beyond simulations and tests in the verification process [5]. Formal methods, which have been widely used in verification of software systems [4], are being explored to reduce the system-level test and evaluation burden. The benefits of employing formal methods include unambiguous specifications of requirements, exhaustive coverage of the scenario space, and a mathematically rigorous process. Notwithstanding these benefits, formal methods are not well-suited to analyze the whole system due to scalability issues and are used only to verify specific elements of a CPS as noted in [6]. An approach employed in software verification to overcome this difficulty, which is also adopted in this work, is compositional verification [7]–[9], whereby the problem of verifying the global system is reduced to a problem of reasoning about the system's components. Ideally, one would like to prove all the component properties purely in the realm of formal methods or using rigorous mathematical tools; however, in reality, analytically proving the properties of a component that comprises a dynamical system can be a formidable task [1]. For this reason, we adopt a hybrid approach in this work, whereby rigorous mathematical tools and Monte Carlo simulations are judiciously used together to verify the multi-agent system. We design the multi-agent system to have a component-based structure such that most of the components can be verified using rigorous mathematical tools, thereby restricting the use of simulations to a few components where such tools are intractable or conservative.

When verifying a safety-critical system, one is interested in verifying properties concerning the safety, security, and reliability of the system. By adopting a formal approach, we incorporate mathematically rigorous methods from the start of the design process and design the multi-agent system to satisfy a certain global system property. This verified design approach affords substantial gains in reducing the certification time and cost. Herein, we employ the compositional approach proposed in [10] to perform verified design of an unmanned multi-aircraft system. As in other compositional verification approaches, we express the component specifications in an assumption/guarantee style, whereby the component guarantees a property provided that its inputs satisfy certain assumptions. The global system property that we are interested in verifying pertains to the safety and performance of the multi-aircraft system.

The motivation for the case study considered in this paper stems from the need to design a defender system that can overcome the security threats to critical infrastructures posed by an adversary using a small UAS [11]. The case study involves design of an unmanned multi-aircraft system that is tasked to track and compromise an aerial moving target, which is another fixed-wing UAS. An operation of this kind typically involves multiple stages: i) identifying and classifying the aerial intruder using ground-based systems, ii) directing the defenders to intercept the encroacher, and iii) deploying countermeasures to compromise the encroacher. In this work, we focus on the third stage of the operation. Many techniques have been proposed to neutralize and take control of an aerial encroacher, such as capturing drones using nets [12], [13], destroying UAS with interceptor drones or firearms, jamming the encroacher's GPS signal or the radio control link [14], and disabling the encroacher using sonic waves [15] or electromagnetic signals [16]. In many situations, it may be necessary to gather more information about the encroacher, such as its system design, sensor suite, and autopilot hardware, instead of just destroying it. In such a scenario, it is desirable to neutralize the threat from the encroacher and get access to it without imparting physical damage, for instance, using a system that employs the method proposed in [15] or [16]. We assume that such a system is present onboard each defender. However, for any of these methods to be successful, the defender is required to be in close proximity to the encroacher for a certain period of time.

Although the exact capabilities of an encroacher are difficult to ascertain, we assume appropriate bounds on the velocity, acceleration, and maneuvering capabilities of the encroacher based on data available from the Small Aircraft Flight Encounters (SAFE) repository [17]. In order to have meaningful engagement between the encroacher and the defenders, the defenders' capabilities are considered to be superior to those of the encroacher. For the countermeasures to be successful, the designed unmanned multi-aircraft system must satisfy the following desired global system property: at the minimum one defender is within a prespecified distance from the encroacher throughout the mission, and each defender is at least a certain distance away from the encroacher and the other defenders. The first part of the system property ensures a successful takeover of the encroacher, and the second part guarantees that the defenders do not collide with each other or with the encroacher.

The multi-aircraft system is composed of two defender fixed-wing UAS. To make effective use of the compositional approach, the components of the multi-aircraft system are designed to be modular and to have properties that can be composed to achieve the global system property. Specifically, we adopt a virtual-vehicle-based approach where each defender follows a virtual vehicle. The reference (virtual vehicle) trajectory is then tracked by a low-level controller. This internal architecture enables one to decompose the component property into two parts, namely, (i) close tracking and collision avoidance between the virtual vehicle and the encroacher and (ii) close tracking between the defender and the virtual vehicle. We use results from Euclidean geometry and formal methods to prove part (i), while relying on Monte Carlo simulations to verify part (ii). Once the components of the unmanned multi-aircraft system are designed, we formally express the component specifications and their

properties, the environmental assumptions, and the global system property in the temporal logic of actions (TLA) formal language. For the purposes of verification, we abstract the multi-aircraft system as a two-component system with the components being the two defenders. The abstracted system interacts with its environment, which comprises the encroacher and exogenous disturbances in the form of sensor noise, wind, and atmospheric turbulence. The task of verifying the global system property is reduced to the problem of verifying local properties of the two components provided that the components satisfy certain hypotheses. One of these hypotheses addresses the issue of soundness, as it is important to ensure that the potentially circular reasoning necessitated by the compositional approach is sound. The formal methods supplement to DO-178C, namely, the report DO-333 [18], states that a formal method must never produce a result which may not be true. The compositional approach in [10] ensures soundness of circular reasoning by imposing additional constraints on the component specifications. Therefore, once the local component properties are verified, we finally provide formal proofs based on the decomposition theorem of [10] to show that the composition of the components implements the desired global system property.

Contributions

This work demonstrates how compositional reasoning can be effectively used for the design and reliability analysis of complex multi-agent CPS, such as the unmanned multi-aircraft system considered herein. The specific contributions are highlighted below:

1) We show how the components of the multi-aircraft system can be designed to be modular such that the component properties can be composed to obtain the global system property.
2) By leveraging the modularity of the components, we demonstrate how rigorous mathematical tools can be used to prove certain local properties of the components, thereby minimizing the use of simulations to verify local component properties.
3) Although the compositional approach itself is adopted from [10], it is employed here in a completely different application domain to increase the reliability of an unmanned multi-aircraft system. Herein, we deal with components that consist of an uncertain dynamical system, a motion planner, and a robust controller interacting with each other. We show how these components can be specified and their properties verified in TLA.

The problem of designing an unmanned multi-aircraft system and verifying that it satisfies a global system property bears a number of challenges, and the hybrid approach adopted in this work is an attempt towards making the verification process rigorous, which would thereby lead to increased reliability of the system.

The outline of the paper is as follows: Section II gives a brief overview of the compositional approach used in this work; Section III provides a short description of the TLA formal language and the UAS equations of motion; Section IV presents the case study and demonstrates how component-based reasoning can be used to verify the global system property; and Section V gives some concluding remarks.

II. ADOPTED APPROACH

This section presents a summary of the compositional approach used in this work. Compositional reasoning is a powerful approach that is used in the verification of complex software systems [10], [19], [20], whereby the onerous task of verifying the whole system is broken down into simpler subtasks. For instance, given a system with specification M and a global property P, the task of verifying that M implements P, denoted by $M \Rightarrow P$, is simplified using the compositional approach by abstracting M as a composition of N interconnected components, each having a desired property P_i such that $\bigwedge_{i=1}^{N} P_i \Rightarrow P$. The operational environment of a component plays an important role in determining the properties that a component satisfies. In view of this, the component specifications in a compositional reasoning approach are written in an assumption/guarantee style, i.e., under an appropriate environmental assumption E_i, the specification M_i of component i implements the property P_i, written as the formula $E_i \wedge M_i \Rightarrow P_i$.

When there are interactions between the components, the environmental assumption of one component could be drawn from the property of another component, thereby leading to circular reasoning. In general, circular reasoning is not sound; to make it sound, additional constraints need to be placed on the behavior of the components. This typically involves proving a formula slightly stronger than $E_i \wedge M_i \Rightarrow P_i$ for each component. In the compositional approach presented in [10] and adopted in this work, one is required to show that when the environmental assumption E_i fails to hold at a specific step, the component M_i still implements its property P_i until the next step. In [10], all the component and system specifications are written in TLA; a brief overview of this language is provided in Section III-A.

In the case of software systems, formulas of the form $E_i \wedge M_i \Rightarrow P_i$ are proved using tools from formal methods such as model checking and theorem proving. However, for physical systems like the one considered in this work, it is a formidable task to prove all the component properties using tools from formal methods due to state-space explosion. Although tools from robust control such as integral quadratic constraint (IQC) theory [21] can potentially be used to prove properties of physical systems such as UAS, the tools currently available lead to conservative results. Research is currently underway in our group to reduce the conservativeness of the IQC approach, which could pave the way for its use in a future work. In this paper, we utilize other mathematical tools, such as results from Euclidean geometry, to prove certain component properties. For properties that are difficult to be formally proved, we make use of focused Monte Carlo simulations to verify them. Once the local properties are

verified, the global system property is reasoned using the decomposition theorem of [10].

In addition to simplifying the verification task, the compositional approach adopted in this work also affords other benefits that are significant for safety-critical systems. First, by explicitly specifying the assumptions in a formal language, any ambiguity in the assumptions is removed. Second, the use of component-based reasoning and the assumption/guarantee style of specifications enables one to pinpoint the source of the problem in the case of a failure. Third, this approach provides the ability to describe and reason about the system in varying levels of granularity, thereby enabling one to transfer high-level component properties down to the level of executable code or the programmable hardware. Finally, breaking down the verification task into smaller manageable subtasks also paves the way for the use of automated verification tools, such as theorem proving and model checking, for proving certain local component properties.

III. PRELIMINARIES

A. TEMPORAL LOGIC OF ACTIONS

This section presents a brief overview of TLA. For a more elaborate discussion, the interested reader is referred to [22]. A *state* in TLA is a mapping from a set of variables to a set of values. A TLA formula is interpreted in terms of *behaviors*, where a behavior is an infinite sequence of states. Let $\sigma \triangleq (s_0, s_1, \ldots)$ denote a behavior, where s_i is the i^{th} state. The finite sequence $(s_0, s_1, \ldots, s_{i-1})$ is called the i^{th} *prefix* of σ. There are two kinds of variables in TLA, namely, *rigid* variables and *flexible* variables. If the value of a particular variable is fixed but possibly unknown, then that variable is called a rigid variable. If a variable is not rigid, then it is a flexible variable. Although a state in TLA assigns values to all the variables in the universe, when writing a state we restrict our attention only to the variables relevant to the specification. The variables in TLA are not restricted to any specific type, and they could represent scalars, vectors, sequences of vectors, matrices, etc.

A *predicate* in TLA refers to a Boolean-valued expression that contains variables and constant symbols; for instance, $y = 1$ is a predicate. A *state function*, on the other hand, is a non-Boolean valued expression formed from variables and constant symbols; an example of a state function is $(1 + 0.01\alpha)y$. Constant symbols are either numerals or entities that have a fixed value. When writing the specifications, we differentiate constant symbols from TLA variables by using a special font for the constant symbols. For instance, an example of a constant symbol is $\mathtt{D_a}$. A TLA formula is constructed from state functions using the standard Boolean operators ($\neg$, $\vee$) and three other operators, denoted by $'$, $\Box$, and $\exists$, that are described in the subsequent paragraphs.

The current state and the next state in TLA are denoted by unprimed and primed variables, respectively. An *action*, which represents an atomic operation, is a Boolean-valued expression formed from primed variables, unprimed variables, and constant symbols. An action $y' = y+1$ evaluates to TRUE for a pair of states s and t if and only if the value of $y+1$ in state s is equal to the value of y in state t. Given a tuple of variables denoted by f, the action $f' = f$ is written as $Unch(f)$ in the sequel. A pair of states $\langle s, t\rangle$ that satisfies the action $\mathcal{N}$ is called an $\mathcal{N}$ *step*. For an action $\mathcal{N}$ and a state s, *Enabled* $\mathcal{N}$ is a predicate that evaluates to TRUE at the state s if and only if it is possible to take an $\mathcal{N}$ step starting from s. For instance, consider the action $\mathcal{N} \triangleq (y' = 0.01y + u) \wedge (u \geq 0)$ and the state pair $\langle s, t\rangle$, where $s \triangleq (y \mapsto 100, u \mapsto 3, \ldots)$ and $t \triangleq (y \mapsto 4, u \mapsto -2, \ldots)$. *Enabled* $\mathcal{N}$ in this case evaluates to TRUE and FALSE for the states s and t, respectively. The state pair $\langle s, t\rangle$ in this example is an $\mathcal{N}$ step. Given an action $\mathcal{N}$ and a state function f, $[\mathcal{N}]_f$ and $\langle\mathcal{N}\rangle_f$ are defined as $[\mathcal{N}]_f \triangleq \mathcal{N} \vee Unch(f)$ and $\langle\mathcal{N}\rangle_f \triangleq \mathcal{N} \wedge \neg Unch(f)$.

As in standard temporal logic, $\Box\mathcal{N}$ is TRUE of a behavior if and only if each step of the behavior is an $\mathcal{N}$ step, where a step refers to a state pair that comprises two consecutive states of a behavior. In the sequel, we will sometimes refer to an execution of a component's action also as a step. A formula P is said to be *valid*, denoted as $\models P$, if and only if P is TRUE for every behavior. When specifying a system in TLA, in addition to specifying the allowable behaviors, it is also desirable to specify the following requirement: if a certain operation is possible, then the system must eventually execute it. This type of condition is called a fairness condition, which in TLA is expressed either as *weak-fairness* or *strong-fairness*. A weak-fairness condition specifies that an action that is almost always enabled should be executed infinitely often. A strong-fairness condition, on the other hand, specifies that an action which is infinitely often enabled should be executed infinitely often. In TLA, the weak-fairness and strong-fairness conditions are specified by the formulas $WF_f(\mathcal{N}) \triangleq (\Box\Diamond\langle\mathcal{N}\rangle_f) \vee (\Box\Diamond\neg Enabled\langle\mathcal{N}\rangle_f)$ and $SF_f(\mathcal{N}) \triangleq (\Box\Diamond\langle\mathcal{N}\rangle_f) \vee (\Diamond\Box\neg Enabled\langle\mathcal{N}\rangle_f)$, respectively. The operator $\Diamond$ is shorthand for $\neg\Box\neg$. For a flexible variable y and a formula P, $\exists y : P$ states that a sequence of values for y can be chosen such that P holds. y is called an *internal* variable of the formula $\exists y : P$.

A TLA formula in canonical form is written as $\exists y : Init \wedge \Box[\mathcal{N}]_f \wedge F$, where *Init* and F represent a state predicate and a conjunction of fairness conditions, respectively. The semantics of the formula is as follows: there exists a sequence of values for y such that *Init* is TRUE for the initial state, and every step of the behavior is an $\mathcal{N}$ step or leaves the state function f unchanged, and F holds. A *safety property* is a formula that is satisfied by a behavior σ if and only if the formula is satisfied by every prefix of σ. The *closure* of a TLA formula M, denoted by $\mathcal{C}(M)$, is the strongest safety property such that $\models M \Rightarrow \mathcal{C}(M)$. The formula M_{+f} is TRUE for a behavior σ if and only if either σ satisfies M, or there exists i such that M holds for the first i states of σ and f remains unchanged from the $(i+1)^{\text{th}}$ state onwards.

Given a system, let $E \triangleq Init_E \wedge [\mathcal{N}_E]_{\langle v_E \rangle}$, $M \triangleq Init_M \wedge [\mathcal{N}_M]_{\langle v_M \rangle} \wedge F_M$, and $P \triangleq Init_P \wedge [\mathcal{N}_P]_{\langle v_P \rangle}$ denote TLA formulas pertaining to the environmental assumption, the system specification, and the system property, respectively. Then, proving that the system specification M implements the system property P under the environmental assumption E amounts to showing $\models E \wedge M \Rightarrow P$. In TLA, the procedure to prove such a formula is broken down into proving the following two formulas: $Init_E \wedge Init_M \Rightarrow Init_P$, $[\mathcal{N}_E]_{\langle v_E \rangle} \wedge [\mathcal{N}_M]_{\langle v_M \rangle} \Rightarrow [\mathcal{N}_P]_{\langle v_P \rangle}$. In the sequel, the proof technique for proving a formula of the form $E \wedge M \Rightarrow P$ using the axioms and proof rules of TLA is referred to as step simulation. The complete set of axioms and proof rules of TLA can be found in [22].

Consider a system with specification M, environmental assumption E, and global property P. If it is possible to abstract this system as a composition of interacting components, then the compositional approach described in Section II may be employed to simplify the task of verifying the global property P (i.e., proving the assertion $\models E \wedge M \Rightarrow P$). Let E_i, M_i, and P_i denote component i's environmental assumption, specification, and property, respectively, where $\bigwedge_{i=1}^{n} P_i \Rightarrow P$ and n denotes the number of components. The following theorem from [10] will be used to verify the assertion $\models E \wedge \bigwedge_{i=1}^{n} M_i \Rightarrow \bigwedge_{i=1}^{n} P_i$, which constitutes the first step in verifying the global system property P.

Theorem 1: (Decomposition Theorem) If, for $i = 1, \ldots, n$,

(1) $\models \mathcal{C}(E) \wedge \bigwedge_{j=1}^{n} \mathcal{C}(P_j) \Rightarrow E_i$,
(2) (a) $\models \mathcal{C}(E_i)_{+v} \wedge \mathcal{C}(M_i) \Rightarrow \mathcal{C}(P_i)$,
    (b) $\models E_i \wedge M_i \Rightarrow P_i$,
(3) *v is a tuple that includes all the free variables of P_i, then*
(i) $\models \mathcal{C}(E)_{+v} \wedge \bigwedge_{j=1}^{n} \mathcal{C}(M_j) \Rightarrow \bigwedge_{j=1}^{n} \mathcal{C}(P_j)$,
(ii) $\models E \wedge \bigwedge_{j=1}^{n} M_j \Rightarrow \bigwedge_{j=1}^{n} P_j$.

B. UNMANNED AIRCRAFT SYSTEM MODEL

In this section, we describe the dynamic model of the defenders. The dynamic model pertains to the commercially available Senior Telemaster radio-controlled aircraft from Hobby Express [23]. For an elaborate discussion on the rigid-body equations of motion for a fixed-wing aircraft, the interested reader is referred to any standard textbook on flight dynamics and control; see, for instance, [24]. To represent the UAS motion, we need two reference frames, namely, the Earth-fixed inertial reference frame, also known as the NED frame, and the body-fixed reference frame. The origin of the NED frame is on the surface of the Earth, and it has components which point along the north, the east, and the downward directions, respectively. The position vector from the origin of the NED frame to the nominal aircraft center of gravity is defined as $\boldsymbol{p} = [X, Y, Z]^T$, where X, Y, and Z denote the north, east, and downward components, respectively. The origin of the body frame is at the aircraft center of gravity, and it has components that point towards the aircraft nose, right wingtip, and downwards, respectively. The UAS has three control surfaces and one propulsive unit. The three control surfaces are the aileron, elevator, and rudder, and they are used to control the UAS motion about the x, y, and z axes of the body frame, respectively. An electric motor-driven propeller forms the propulsive unit.

Let the roll, pitch, and yaw Euler angles that define the orientation of the body frame with respect to the inertial frame be denoted as $\boldsymbol{\lambda} = [\phi, \theta, \psi]^T$. We denote the linear velocity of the body frame with respect to the inertial frame expressed in the body frame as $\boldsymbol{v} = [u, v, w]^T$, where u, v, and w are the three components of the UAS inertial velocity vector. In a similar manner, the angular velocity of the body frame is denoted as $\boldsymbol{\omega} = [p, q, r]^T$, where p, q, and r are the roll, pitch, and yaw rates of the UAS. The state vector of the UAS can then be defined as $\boldsymbol{x} = [\boldsymbol{v}^T, \boldsymbol{\omega}^T, \boldsymbol{\lambda}^T, \boldsymbol{p}^T]^T$. The flight path angle, denoted by η, is defined as $\eta = \arctan(w/u)$.

Since the UAS is symmetric about the xz-plane of the body frame, we have $I_{yz} = I_{zy} = I_{xy} = I_{yx} = 0$. In addition, the remaining two cross products of inertia, I_{xz} and I_{zx}, are assumed to be negligibly small. The other inertia terms are estimated through compound pendulum tests resulting in $I_x = 1.32\,\mathrm{kg/m^2}$, $I_y = 1.57\,\mathrm{kg/m^2}$, and $I_z = 1.87\,\mathrm{kg/m^2}$ [25]. Let the input vector of the UAS be denoted as $\boldsymbol{\delta} = [\delta_E, \delta_A, \delta_R, \delta_T]^T$, where δ_E, δ_A, and δ_R are the elevator, aileron, and rudder deflections, respectively, and δ_T is the throttle input. These inputs are expressed as pulse-width modulated (PWM) signals. The net external force and moment acting on the UAS expressed in the body frame are denoted as $\boldsymbol{f}(\bar{\boldsymbol{v}}, \boldsymbol{\omega}, \boldsymbol{\delta})$ and $\boldsymbol{m}(\bar{\boldsymbol{v}}, \boldsymbol{\omega}, \boldsymbol{\delta})$, respectively, where $\bar{\boldsymbol{v}}$ is the linear velocity of the UAS relative to the wind. The net external force and moment comprise forces and moments due to aerodynamics and propulsion. These forces and moments are modeled as in [25]. The nonlinear differential equations that govern the UAS motion can now be written as

$$\dot{\boldsymbol{v}} = m^{-1}\boldsymbol{f}(\bar{\boldsymbol{v}}, \boldsymbol{\omega}, \boldsymbol{\delta}) + \boldsymbol{g} - \boldsymbol{\omega} \times \boldsymbol{v}$$
$$\dot{\boldsymbol{\omega}} = \boldsymbol{J}_b^{-1}\boldsymbol{m}(\bar{\boldsymbol{v}}, \boldsymbol{\omega}, \boldsymbol{\delta}) - \boldsymbol{J}_b^{-1}(\boldsymbol{\omega} \times \boldsymbol{J}_b\boldsymbol{\omega})$$
$$\dot{\boldsymbol{\lambda}} = \boldsymbol{E}(\phi, \theta)\boldsymbol{\omega}, \quad \dot{\boldsymbol{p}} = \mathcal{R}_{Ib}(\boldsymbol{\lambda})\boldsymbol{v}, \text{ where}$$
$$\boldsymbol{E}(\phi, \theta) = \begin{bmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix} \text{ and}$$
$$\boldsymbol{J}_b = \begin{bmatrix} I_x & 0 & 0 \\ 0 & I_y & 0 \\ 0 & 0 & I_z \end{bmatrix}.$$

$\mathcal{R}_{Ib}$ is the rotation matrix from the body frame to the NED frame, $\boldsymbol{g}$ is the gravitational acceleration vector, $\boldsymbol{J}_b$ is the inertia matrix, and $m = 5.71\,\mathrm{kg}$ is the mass of the UAS.

Each control surface is actuated by a servomotor that is modeled as a second-order system with natural frequency and damping ratio of 13.7 rad/s and 0.67, respectively. The second-order servo model maps the control surface command δ_{j_c} to the corresponding control surface deflection δ_j for $j \in \{E, A, R\}$.

IV. CASE STUDY

The case study concerns the verified design of two fixed-wing UAS, the defenders, whose joint objective is to maintain close formation with another fixed-wing UAS, the encroacher, to enable the deployment of countermeasures. The following subsections provide the specifics of the design, including the assumptions made and the specifications/properties of the components and overall system.

A. PROBLEM STATEMENT

This subsection lists the assumptions made on the capabilities of the encroacher and defenders, along with the values used for several design parameters, and provides the initial setup and desired global system property.

1) ENCROACHER

We make some broad assumptions on the capabilities of the encroacher based on the SAFE data repository [17]. Specifically, the maximum speed, minimum speed, and maximum vertical speed of the encroacher are assumed to be 20 m/s, 10 m/s, and 10 m/s, respectively. To ensure that the encroacher's trajectory is not physically unrealistic, we assume that the bounds on the magnitudes of the accelerations along the x, y, and z axes of the NED frame are 7 m/s^2, 7 m/s^2, and 3 m/s^2, respectively, where these values are again obtained from the SAFE data repository. We assume the maneuvering capabilities of the encroacher are such that the minimum radius of curvature of its trajectory is at least 80 m.

2) DEFENDERS

We assume that the defenders can move faster than the encroacher with a maximum speed of 25 m/s. Ideally, one defender could be sufficient to compromise the encroacher. However, with just one defender, a high-performance controller is required for the defender to be able to closely track the encroacher. Based on simulation studies with robust linear $\mathcal{H}_\infty$ controllers, we observed that to achieve good position tracking, the defender should have access to the attitude measurements of the encroacher, namely, the roll, pitch, and yaw Euler angles. Since obtaining accurate attitude measurements is difficult in practice [26], compromising the encroacher using just one defender is a challenging task. So, we consider two defenders with similar capabilities that cooperate to achieve this task.

In this work, we adopt a modular approach wherein each defender has a motion planner that generates the reference (virtual vehicle) trajectory and a low-level controller that tracks the reference trajectory. The controller, described in Section IV-C, does not require the encroacher's attitude measurements and is found to closely track the virtual vehicle when the vertical speed of the virtual vehicle does not exceed 5 m/s. Thus, one of the requirements of the motion planner is to plan a trajectory that does not demand a vertical speed greater than 5 m/s even though the encroacher can move at vertical speeds of up to 10 m/s. This requirement can be afforded because of the modular architecture of the system design, where the desired global system property can be decomposed into (1) proximity and collision avoidance properties between the virtual vehicles and the encroacher and (2) a proximity property between each defender and its virtual vehicle. Imposing the 5 m/s limit on the virtual vehicle's vertical speed will enable the design of a low-level controller that achieves property (2), and as will be seen in Section IV-B, the designed motion planners will achieve properties (1) while accommodating this imposed restriction.

3) SYSTEM DESIGN PARAMETERS

There are several system design parameters that need to be specified.

- The safe separation distance, $\mathtt{D_s}$, is the minimum allowable distance between the centers of gravity of two aircraft. Its value is chosen such that an aircraft with a wing span of 8 ft, or 2.44 m (wing span of the UAS considered in this work), has at least a clearance of 1 m with other aircraft of comparable wing span. The reason we only consider the wing span is that the tip of the wing is the farthest point from the center of gravity for such small aircraft. Based on this criterion, the value of $\mathtt{D_s}$ is chosen as 4 m.
- The maximum attack distance, $\mathtt{D_a}$, is the farthest distance a defender can be from the encroacher while still being able to launch a successful attack. This distance depends on the attack mechanism and is assumed in this work to be 35 m.
- The mission duration is another design parameter that needs to be set. One of the goals of the defenders is to restrict the vertical motion of the encroacher to be within a vertical band of ± 50 m about its altitude at the start of the mission. This requirement stems from the operating limitations of small UAS as outlined in [27], which states that a small UAS cannot be operated above 400 ft ($\approx$ 122 m) with reference to the uppermost point on any nearby structure. Therefore, to prevent the defenders from exceeding this limit, we require that these defenders restrict the vertical motion of the encroacher to be within a vertical band of ± 50 m about its altitude at the start of the mission. Considering the limit on the vertical speed of the encroacher (10 m/s), the encroacher is capable of exiting this band in about 5 s. Thus, it is imperative that the defenders maintain a persistent attack initially for at least 5 s, in which case the defenders will have a hold on the encroacher and restrict its altitude variation throughout the rest of the operation. The attack can then be sustained, as the proximity and collision avoidance properties discussed in the following will be satisfied. Therefore, for the purposes of this case study, the mission duration is chosen as 6 s.
- Since we adopt a virtual-vehicle-based approach, where each defender tracks a virtual vehicle, we have to specify three more design parameters pertaining to the virtual vehicle: the distance between the encroacher and the

virtual vehicle when its associated defender is carrying out the attack, D_1; the maximum distance between the defender and its virtual vehicle, D_2; and the vertical separation (offset) between the two virtual vehicles, H_{of}. Based on an extensive analysis of the defender's controller, it is observed that the position tracking error between the defender and its virtual vehicle is less than 15 m. We will later see in Section IV-C that the error bound, D_2, is actually 14 m. Consequently, as the maximum attack distance $D_a = 35$ m, at least one of the virtual vehicles should be at a distance of $D_1 \approx 20$ m from the encroacher to ensure a persistent attack. Specifically, we have $D_1 \in [20-\epsilon_1, 20+\epsilon_1]$ for a small positive scalar ϵ_1 that designates the tolerance; it will be convenient to take $\epsilon_1 = 0.35$. Although the vertical separation between a defender's virtual vehicle and the encroacher could be D_1 (consequently resulting in a vertical separation of $2D_1$ between the virtual vehicles), due to the differences in the vertical speed limits between the encroacher and the virtual vehicle, the vertical separation has to be limited to a value less than D_1 so that the motion planner could still generate a feasible reference trajectory for the defender. Thus, the vertical separation, H_{of}, between the virtual vehicles is chosen to be 32 m.

4) INITIAL CONDITIONS

The encroacher is initially assumed to be trapped between the two defenders. The initial conditions are chosen such that they are general enough to account for a wide range of configurations while enabling the proper initialization of the virtual vehicles. Specifically, the motion planners discussed in Section IV-B require the following at the start of the mission: at least one of the virtual vehicles is at a distance of D_1 from the encroacher, the vertical separation between the encroacher and each virtual vehicle is $H_{of}/2$, and the vertical separation between the two virtual vehicles is H_{of}. Additionally, when verifying the proximity property between the defender and its virtual vehicle in Section IV-D, we assume that the initial separation between the defender and its virtual vehicle is no greater than $D_2/2$. With these in mind, we assume the following initial conditions: i) the planar distance between each defender and the encroacher is within the range $D_3 \pm D_s/2$, where D_3 depends on the parameters H_{of} and D_1 and has a value of 12 m; ii) the vertical separation between each defender and the encroacher is within the range $(H_{of} \pm D_s)/2$; iii) the planar distance between the two defenders is less than or equal to D_s; and iv) the vertical separation between the two defenders is within the range $H_{of} \pm D_s$.

5) GLOBAL SYSTEM PROPERTY

To successfully compromise the encroacher, the designed multi-aircraft system must satisfy the following global system property: throughout the mission duration, at least one defender is within a distance of D_a from the encroacher, and each defender is at least at a distance of D_s from the other defender and the encroacher.

We use the compositional approach described in Section II to verify the global system property. Since our intention is to increase the reliability of the designed system by reducing the use of simulations in the verification process, we design the defenders to be modular. Specifically, each defender has a motion planner, described in Section IV-B, that takes the sensed position and velocity of the encroacher as inputs and computes the reference (virtual vehicle) trajectory. The defender then tracks the reference trajectory despite significant atmospheric disturbances using a low-level controller, which is described in Section IV-C. In the sequel, the label "defender 1" is used to designate the defender that starts above the encroacher. The modularity of defender 1 is leveraged in Section IV-D to verify its component property, wherein we formally prove the properties of the motion planner. This component property amounts to guaranteeing that the virtual vehicle of the defender is always at a distance of D_1 from the encroacher when the defender is tracking the encroacher and the distance between the two virtual vehicles is always H_{of}. We then make use of focused Monte Carlo simulations to bound the error between the defender and its virtual vehicle. In a similar manner, we verify analogous assertions for defender 2 in Section IV-E, as both defenders share the same modular design. Once the environmental assumptions, lower-level specifications, component properties, and the global system property are formally specified, we use the decomposition theorem of [10] to show that the component properties are composable and the composition of the components implements the global system property. The interactions among the components of the abstracted multi-aircraft system, its environment, and the encroacher are shown in Fig. 1; the description of the variables in the figure is provided in Table 1. The motion planner is presented next.

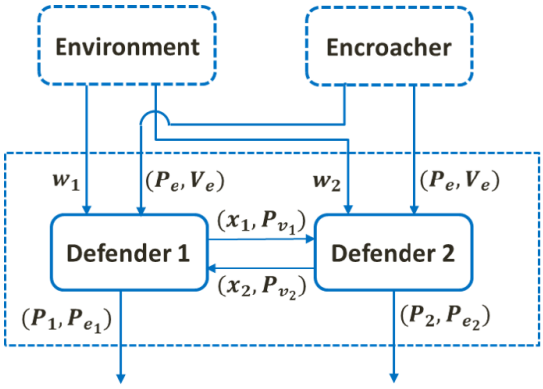

FIGURE 1. The abstracted unmanned multi-aircraft system considered in the case study.

B. MOTION PLANNER

Each defender has its own motion planner that computes the position of its virtual vehicle, i.e., its desired position, while

TABLE 1. Commonly used symbols in the case study.

Symbol	Description
$\mathtt{D_s}$	safe separation distance
$\mathtt{D_a}$	maximum attack distance
$\mathtt{D_1}$	distance between the encroacher and the virtual vehicle when its associated defender is carrying out the attack
$\mathtt{D_2}$	maximum distance between the defender and its virtual vehicle
$\mathtt{D_3}$	parameter used to specify the initial planar distance between a defender and the encroacher
$\mathtt{H_{of}}$	vertical separation between the virtual vehicles of the two defenders
ϵ_1	tolerance
$\mathtt{V_{d_{max}}}$	maximum demanded speed of the defender
$\mathtt{V_{zd_{max}}}$	maximum demanded vertical speed of the defender
$\mathtt{V_{e_{min}}}$	minimum speed of the encroacher
$\mathtt{V_{e_{max}}}$	maximum speed of the encroacher
$\mathtt{V_{ze_{max}}}$	maximum vertical speed of the encroacher
τ	sampling time
Mode_i	mode of operation of defender i
V1	virtual vehicle of defender 1
$\widehat{\mathrm{V2}}$	virtual vehicle of defender 2 as perceived by defender 1
EN	encroacher
P_e	position vector of the encroacher
V_e	velocity vector of the encroacher
P_{e_i}	position of the encroacher as sensed by defender i
V_{e_i}	velocity of the encroacher as sensed by defender i
P_{v_i}	position of defender i's virtual vehicle
$\hat{P}_{v_1}$	position of defender 1's virtual vehicle as estimated by defender 2
$\hat{P}_{v_2}$	position of defender 2's virtual vehicle as estimated by defender 1
P_i	position of defender i
x_i	state vector of defender i
u_i	control vector of defender i
w_i	disturbance vector of defender i
$(x_e(\cdot), y_e(\cdot), z_e(\cdot))$	position coordinates of the encroacher
$(v_{xe}(\cdot), v_{ye}(\cdot), v_{ze}(\cdot))$	velocity components of the encroacher
$(x_{v_i}(\cdot), y_{v_i}(\cdot), z_{v_i}(\cdot))$	position coordinates of defender i's virtual vehicle
$(\hat{x}_{v_i}(\cdot), \hat{y}_{v_i}(\cdot), \hat{z}_{v_i}(\cdot))$	position coordinates of defender i's virtual vehicle as estimated by the other defender

TABLE 2. Values of the parameters used in the case study.

ϵ_1	τ	$\mathtt{D_1}$	$\mathtt{D_2}$	$\mathtt{D_3}$	$\mathtt{H_{of}}$	$\mathtt{D_s}$	$\mathtt{D_a}$
0.35	0.04	$[20-\epsilon_1, 20+\epsilon_1]$	14	12	32	4	35

$\mathtt{V_{d_{max}}}$	$\mathtt{V_{zd_{max}}}$	$\mathtt{V_{e_{min}}}$	$\mathtt{V_{e_{max}}}$	$\mathtt{V_{ze_{max}}}$
25	5	10	20	10

respecting the restrictions on the demanded total speed and vertical speed. During initialization, each defender shares the initial position of its virtual vehicle with the other defender. Beyond this, the two defenders do not communicate with each other. At each time instant, the motion planner of a defender takes the current position and velocity of the encroacher, the current position of its virtual vehicle, the current estimated position of the other virtual vehicle, and its mode of operation as inputs, and returns the position of its virtual vehicle at the next time instant. In addition to computing the position of its virtual vehicle, the motion planner of each defender also estimates the virtual vehicle position of the other defender. There are two modes of operation for a defender; it tracks either the encroacher or the other defender's virtual vehicle as perceived by it. At any time instant, only one defender is tracking the encroacher.

The following notation is used in the sequel. $\mathtt{V_{d_{max}}}$ and $\mathtt{V_{zd_{max}}}$ denote the maximum demanded speed and maximum demanded vertical speed of the defender, respectively. For $i=1,2$, Mode_i denotes the mode of operation of defender i. When $\mathrm{Mode}_i=1$, defender i tracks the encroacher; when $\mathrm{Mode}_i=2$, defender i tracks the other defender. Let $(x_e(k), y_e(k), z_e(k))$, $(x_{v_i}(k), y_{v_i}(k), z_{v_i}(k))$, and $(\hat{x}_{v_i}(k), \hat{y}_{v_i}(k), \hat{z}_{v_i}(k))$ denote the position coordinates of the encroacher, the virtual vehicle of defender i, and the virtual vehicle of defender i as estimated by the other defender, respectively, at time $t_k=k\tau$, where k is any nonnegative integer and $\tau = 0.04\,\mathrm{s}$ is the sampling time. v_{xv_i}, v_{yv_i}, and v_{zv_i} denote the components of the velocity vector of defender i's virtual vehicle. As the two motion planners are similar, we focus on the motion planner of defender 1, which is given in Algorithm 1.

Before describing the algorithm, we specify the initial positions of the virtual vehicles based on those of the encroacher and the defender. Defender 1's virtual vehicle position is chosen as $(x_e(0)+\mathtt{D_3}\sin\theta_{de},\ y_e(0)+\mathtt{D_3}\cos\theta_{de},\ z_e(0)-\mathtt{H_{of}}/2)$, where θ_{de} depends on the positions of the encroacher and the defender at time $t_k=0$ and is defined as $\theta_{de} = \arctan\left((x_{d1}(0)-x_e(0))/(y_{d1}(0)-y_e(0))\right)$, with $(x_{d1}(0),\ y_{d1}(0))$ being the planar position of the defender at time $t_k=0$. Noting that the z-axis is defined positive downwards, the position of defender 2's virtual vehicle is then initialized as $(x_{v_1}(0), y_{v_1}(0), z_{v_1}(0)+\mathtt{H_{of}})$. Based on the chosen initial positions of the two virtual vehicles, the following conditions hold: the virtual vehicles are vertically separated by a distance of $\mathtt{H_{of}}$, at least one of the virtual vehicles is at a distance of $\mathtt{D_1}$ from the encroacher, and the vertical distance between each virtual vehicle and the encroacher is $\mathtt{H_{of}}/2$.

Having initialized the positions of the virtual vehicles, we now describe the motion planning algorithm. If $\mathrm{Mode}_1 = 1$ and the vertical speed of the encroacher is not greater than $\mathtt{V_{zd_{max}}}$ (line 5), then the positions of the virtual vehicles are updated by setting the velocity of defender 1's virtual vehicle to be equal to that of the encroacher (lines 6-7). Alternatively, if the vertical speed of the encroacher is greater than $\mathtt{V_{zd_{max}}}$ (line 12), Algorithm 1 computes the velocity of defender 1's virtual vehicle (lines 17-22) by ensuring that the

limits on the demanded total speed and vertical speed are respected and the distance between the virtual vehicle and the encroacher at the next time instant is $\mathtt{D}_1$ (lines 13 and 16). Then, the updated position of defender 1's virtual vehicle is determined using a first-order approximation (line 24). Based on this information, the updated position of defender 2's virtual vehicle is estimated (line 25). When $\text{Mode}_1 = 2$, the position of defender 1's virtual vehicle is computed from the estimated position of defender 2's virtual vehicle (lines 9 and 27). Since the motion planners of the two defenders are similar, estimating the position of the other defender's virtual vehicle follows the same procedure as before if one knows the initial positions of the virtual vehicles and the position and velocity of the encroacher at each time instant.

In the rest of this section, we formally prove the following results for the motion planner through Lemma 1. When defender 1 is tracking the encroacher ($\text{Mode}_1 = 1$), Algorithm 1 ensures that the separation distance between defender 1's virtual vehicle and the encroacher is always equal to $\mathtt{D}_1$. When $\text{Mode}_1 = 2$, i.e., defender 1 is tracking defender 2, the distance between the encroacher and defender 1's virtual vehicle is guaranteed to be greater than or equal to $\mathtt{D}_1 - \epsilon_1$. Additionally, under both modes of operation, the virtual vehicles of both defenders are vertically separated by a distance of $\mathtt{H}_{\mathtt{of}}$. These results will later be used in Sections IV-D and IV-G to verify the global system property. Before presenting the lemma, we make some additional definitions. $dx(\cdot)$, $dy(\cdot)$, and $dz(\cdot)$ are the differences in the x, y, and z position coordinates between defender 1's virtual vehicle and the encroacher. Similarly, $\hat{dx}(\cdot)$, $\hat{dy}(\cdot)$, and $\hat{dz}(\cdot)$ are the differences in position coordinates between defender 2's virtual vehicle as estimated by Algorithm 1 and the encroacher. We use $\text{dist}(\text{V1}, \text{EN}, k)$ to represent the distance between defender 1's virtual vehicle (V1) and the encroacher (EN) at time t_k, i.e., $\text{dist}(\text{V1}, \text{EN}, k) \triangleq \sqrt{dx(k)^2 + dy(k)^2 + dz(k)^2}$. Similarly, $\text{dist}(\widehat{\text{V2}}, \text{EN}, k)$ is the distance between defender 2's virtual vehicle as perceived by defender 1 ($\widehat{\text{V2}}$) and the encroacher at time t_k. Finally, $\text{dist}(\text{V1}, \widehat{\text{V2}}, k)$ is the distance between V1 and $\widehat{\text{V2}}$ at time t_k. With these definitions, we now state the following result.

Lemma 1: Given the constants and system design parameters as in Table 2, Algorithm 1 *guarantees the following*:

1) *If* $\text{dist}(\text{V1}, \text{EN}, k) = \mathtt{D}_1$, $|dz(k)| \leq \mathtt{H}_{\mathtt{of}}/2$, *and* $\text{Mode}_1 = 1$, *then*
   a) $\text{dist}(\text{V1}, \text{EN}, k+1) = \mathtt{D}_1$
   b) $\text{dist}(\text{V1}, \widehat{\text{V2}}, k+1) = \mathtt{H}_{\mathtt{of}}$
   c) $\text{dist}(\widehat{\text{V2}}, \text{EN}, k+1) \geq \mathtt{D}_1 - \epsilon_1$
2) *If* $\text{dist}(\widehat{\text{V2}}, \text{EN}, k) = \mathtt{D}_1$, $|\hat{dz}(k)| \leq \mathtt{H}_{\mathtt{of}}/2$, *and* $\text{Mode}_1 = 2$, *then*
   a) $\text{dist}(\widehat{\text{V2}}, \text{EN}, k+1) = \mathtt{D}_1$
   b) $\text{dist}(\text{V1}, \widehat{\text{V2}}, k+1) = \mathtt{H}_{\mathtt{of}}$
   c) $\text{dist}(\text{V1}, \text{EN}, k+1) \geq \mathtt{D}_1 - \epsilon_1$.

Proof: The proof is given in the appendix. ■

Algorithm 1: Pseudocode describing the motion planner for defender 1

1: **Inputs**: current sensed position of the encroacher $(x_e(k), y_e(k), z_e(k))$, current sensed velocity of the encroacher $(v_{xe}(k), v_{ye}(k), v_{ze}(k))$, current location of defender 1's virtual vehicle $(x_{v_1}(k), y_{v_1}(k), z_{v_1}(k))$, current location of defender 2's virtual vehicle as estimated by defender 1 $(\hat{x}_{v_2}(k), \hat{y}_{v_2}(k), \hat{z}_{v_2}(k))$, and operational mode ($\text{Mode}_1$)
2: **Outputs**: location of defender 1's virtual vehicle at the next time instant $(x_{v_1}(k+1), y_{v_1}(k+1), z_{v_1}(k+1))$ and location of defender 2's virtual vehicle at the next time instant as estimated by defender 1 $(\hat{x}_{v_2}(k+1), \hat{y}_{v_2}(k+1), \hat{z}_{v_2}(k+1))$
3: **procedure**
4: **if** $|v_{ze}(k)| \leq \mathtt{V}_{\mathtt{zd}_{\max}}$ **then**
5: **if** $\text{Mode}_1 = 1$ **then**
6: $\mu_{v_1}(k+1) = \mu_{v_1}(k) + \tau v_{\mu e}(k)$, where $\mu = x, y, z$
7: $\hat{x}_{v_2}(k+1) = x_{v_1}(k+1)$, $\hat{y}_{v_2}(k+1) = y_{v_1}(k+1)$ $\hat{z}_{v_2}(k+1) = z_{v_1}(k+1) + \mathtt{H}_{\mathtt{of}}$
8: **else if** $\text{Mode}_1 = 2$ **then**
9: $\hat{\mu}_{v_2}(k+1) = \hat{\mu}_{v_2}(k) + \tau v_{\mu e}(k)$, where $\mu = x, y, z$
10: $x_{v_1}(k+1) = \hat{x}_{v_2}(k+1)$, $y_{v_1}(k+1) = \hat{y}_{v_2}(k+1)$ $z_{v_1}(k+1) = \hat{z}_{v_2}(k+1) - \mathtt{H}_{\mathtt{of}}$
11: **else**
12: **if** $\text{Mode}_1 = 1$ **then**
13: $dx(k) = x_{v_1}(k) - x_e(k)$, $dy(k) = y_{v_1}(k) - y_e(k)$ $dz(k) = z_{v_1}(k) - z_e(k)$
14: **else if** $\text{Mode}_1 = 2$ **then**
15: $dx(k) = \hat{x}_{v_2}(k) - x_e(k)$, $dy(k) = \hat{y}_{v_2}(k) - y_e(k)$ $dz(k) = \hat{z}_{v_2}(k) - z_e(k)$
16: Let $v_{zv_1} = \mathtt{V}_{\mathtt{zd}_{\max}} \operatorname{sgn}(v_{ze}(k))$, $(c_{1x}, c_{1y}) = (0, 0)$ $(c_{2x}, c_{2y}) = (v_{xe}(k) - dx(k)/\tau, v_{ye}(k) - dy(k)/\tau)$ $r_1 = \sqrt{\mathtt{V}_{\mathtt{zd}_{\max}}^2 - v_{zv_1}^2}$, $d_{12} = \sqrt{c_{2x}^2 + c_{2y}^2}$, and $r_2 = \sqrt{(\mathtt{D}_1/\tau)^2 - (v_{zv_1} - v_{ze}(k) + dz(k)/\tau)^2}$
17: Let $(v_{xv}^{(1)}, v_{yv}^{(1)})$ and $(v_{xv}^{(2)}, v_{yv}^{(2)})$ denote the points of intersection of the two circles C_1 and C_2 with centers at (c_{1x}, c_{1y}) and (c_{2x}, c_{2y}), and radii r_1 and r_2
18: For $i = 1, 2$, let $\chi_{N_i} = [v_{xv}^{(i)}, v_{yv}^{(i)}]\,[v_{xe}(k), v_{ye}(k)]^T$ $\chi_{D_i} = \|[v_{xv}^{(i)}, v_{yv}^{(i)}]\| \, \|[v_{xe}(k), v_{ye}(k)]\|$, and $\chi_i = \arccos(\chi_{N_i}/\chi_{D_i})$
19: **if** $\chi_1 \leq \chi_2$ **then**
20: $v_{xv_1} = v_{xv}^{(1)}$ and $v_{yv_1} = v_{yv}^{(1)}$
21: **else**
22: $v_{xv_1} = v_{xv}^{(2)}$ and $v_{yv_1} = v_{yv}^{(2)}$
23: **if** $\text{Mode}_1 = 1$ **then**
24: $\mu_{v_1}(k+1) = \mu_{v_1}(k) + \tau v_{\mu v_1}$, where $\mu = x, y, z$
25: $\hat{x}_{v_2}(k+1) = x_{v_1}(k+1)$, $\hat{y}_{v_2}(k+1) = y_{v_1}(k+1)$ $\hat{z}_{v_2}(k+1) = z_{v_1}(k+1) + \mathtt{H}_{\mathtt{of}}$
26: **else if** $\text{Mode}_1 = 2$ **then**
27: $\hat{\mu}_{v_2}(k+1) = \hat{\mu}_{v_2}(k) + \tau v_{\mu v_1}$, where $\mu = x, y, z$
28: $x_{v_1}(k+1) = \hat{x}_{v_2}(k+1)$, $y_{v_1}(k+1) = \hat{y}_{v_2}(k+1)$ $z_{v_1}(k+1) = \hat{z}_{v_2}(k+1) - \mathtt{H}_{\mathtt{of}}$

Since the algorithm for defender 2's motion planner is similar to Algorithm 1, we can prove a result analogous to Lemma 1 for defender 2. Next, we describe the controller.

C. CONTROLLER DESCRIPTION

The purpose of the low-level controller is to enable the defender to track its virtual vehicle whose motion is determined by the motion planner. Both defenders use the controller described in this section. The conventional approach to design such a controller is to consider the position of the virtual vehicle as a command signal to the system and then design a controller that minimizes the error between the UAS position and the virtual vehicle position. However, it is observed that the controller designed using this approach performs poorly for virtual vehicle trajectories with significant altitude variations. Therefore, in this work, we adopt a reference tracking control approach, where the controller is designed to track a reference trajectory that specifies reference values for the aircraft state and control input at every time instant. This control approach is found to result in good tracking performance for virtual vehicle trajectories with vertical speeds not exceeding 5 m/s. The procedure used to compute the reference values and the control design process are described next.

The physical process of the defender is governed by the UAS equations of motion given in Section III-B. The state, control input, and disturbance input of the UAS are defined as $\boldsymbol{x}^c = [\boldsymbol{v}^T, \boldsymbol{\omega}^T, \boldsymbol{\lambda}^T, \boldsymbol{p}^T, \boldsymbol{x}_a^T]^T$, $\boldsymbol{u}^c = [\delta_{E_c}, \delta_{A_c}, \delta_{R_c}, \delta_T]^T$, and $\boldsymbol{w}^c = [\boldsymbol{v}_w^T, \boldsymbol{w}_m^T]^T$, respectively. $\boldsymbol{v}_w$ is the wind velocity, and $\boldsymbol{w}_m$ denotes the measurement noise, defined as $\boldsymbol{w}_m = [m_p, m_q, m_r, m_\phi, m_\theta, m_\psi, m_{V_a}, m_x, m_y, m_z, m_\eta]^T$. $\boldsymbol{x}_a$ is composed of the actuators' states. The measurement output is given by $\boldsymbol{y}^c = [\boldsymbol{\omega}^T, \boldsymbol{\lambda}^T, V_a, \boldsymbol{p}^T, \eta]^T$, where V_a denotes the airspeed and is defined as $V_a = \sqrt{(\boldsymbol{v}-\boldsymbol{v}_w)^T(\boldsymbol{v}-\boldsymbol{v}_w)}$. The vectors $\boldsymbol{x}^c(t)$, $\boldsymbol{u}^c(t)$, $\boldsymbol{y}^c(t)$, and $\boldsymbol{w}^c(t)$ are real and have dimensions 18, 4, 11, and 14, respectively. The performance output $\boldsymbol{z}^c$ consists of weighted functions of state and control variables that we would like to control well, and is chosen as

$$\begin{aligned}\boldsymbol{z}^c = [&0.01\,V_a,\ 0.1\,p,\ 0.1\,q,\ 0.1\,r,\ 0.5\phi,\ 0.5\theta,\ 0.4\psi,\ 0.8\,X,\\ &0.3\,Y,\ 0.3\,Z,\ 0.5\eta,\ 0.7\delta_E,\ 0.6\delta_A,\ 0.6\delta_R,\ 0.7\delta_T]^T.\end{aligned}$$

The state and output equations can be concisely written as $\dot{\boldsymbol{x}}^c = \boldsymbol{f}(\boldsymbol{x}^c, \boldsymbol{w}^c, \boldsymbol{u}^c)$, $\boldsymbol{z}^c = \boldsymbol{g}(\boldsymbol{x}^c, \boldsymbol{w}^c, \boldsymbol{u}^c)$, and $\boldsymbol{y}^c = \boldsymbol{h}(\boldsymbol{x}^c, \boldsymbol{w}^c)$.

Since a UAS is subjected to external disturbances such as atmospheric turbulence, steady wind, and sensor noise during its operation, a controller that provides good performance despite these disturbances is desired. Although different control approaches, such as nonlinear control, adaptive control, and proportional-integral-derivative (PID) control, could be used to design the controller, a robust control approach is adopted in this work. The performance guarantees against exogenous disturbances provided by robust controllers make them a desirable choice for this application. The control design is based on a linear system model, whereby the nonlinear state and output equations are linearized about a trim reference trajectory to obtain a linear time-invariant (LTI) system. A trim reference trajectory is computed from an aircraft trim, which denotes a flight condition where the net forces and moments acting on the aircraft are zero. Examples of trim flight include straight and level flight and circular flight at constant altitude. In this work, we linearize the nonlinear state and output equations about a straight and level trim trajectory. The trim state and control input are obtained by solving the following set of equations using the `MATLAB` function `fsolve`: $\dot{\boldsymbol{v}} = 0$, $\dot{\boldsymbol{\omega}} = 0$, $\dot{\boldsymbol{\lambda}} = 0$, and $\dot{Z} = 0$ with $\boldsymbol{w}^c = 0$ and $V_a = 15$ m/s. The trim values are provided in Table 3.

TABLE 3. Trim state and control input pertaining to straight and level flight.

p	q	r	u	v	w	ϕ	θ
0	0	0	15	0	-3.7×10^{-3}	-3.2×10^{-2}	-2×10^{-4}

Z	δ_E	δ_A	δ_R	δ_T
-630	1.14×10^{-1}	-9.9×10^{-3}	2.5×10^{-2}	4.7×10^{-1}

Units: p, q, r in rad/s; u, v, w in m/s; ϕ, θ in rad; Z in m; $\delta_E, \delta_A, \delta_R, \delta_T$ in ms (PWM)

Let $(\boldsymbol{x}_{tr}, \boldsymbol{u}_{tr}, \boldsymbol{w}_{tr})$ denote the trim reference trajectory corresponding to the straight and level trim given in Table 3, where $\boldsymbol{w}_{tr} = \boldsymbol{0}$. Note that Table 3 gives the trim values for some of the state variables; the rest can be deduced based on the initial position and heading angle of the aircraft. The values provided, however, are the only ones needed to obtain the linearized model. Linearizing the functions $\boldsymbol{f}(\cdot)$, $\boldsymbol{g}(\cdot)$, and $\boldsymbol{h}(\cdot)$ about $(\boldsymbol{x}_{tr}, \boldsymbol{u}_{tr}, \boldsymbol{w}_{tr})$ and performing Euler discretization of the resulting continuous-time LTI system using a sampling time of $\tau = 0.04$ s result in the following discrete-time LTI system:

$$\begin{bmatrix}\bar{\boldsymbol{x}}(k+1)\\ \bar{\boldsymbol{z}}(k)\\ \bar{\boldsymbol{y}}(k)\end{bmatrix} = \begin{bmatrix}\boldsymbol{A} & \boldsymbol{B}_1 & \boldsymbol{B}_2\\ \boldsymbol{C}_1 & \boldsymbol{D}_{11} & \boldsymbol{D}_{12}\\ \boldsymbol{C}_2 & \boldsymbol{D}_{21} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\bar{\boldsymbol{x}}(k)\\ \boldsymbol{w}(k)\\ \bar{\boldsymbol{u}}(k)\end{bmatrix},\ \bar{\boldsymbol{x}}(0) = \boldsymbol{0} \quad (1)$$

where $\boldsymbol{w}(k) = \boldsymbol{w}^c(k\tau)$ and $\bar{\boldsymbol{\mu}}(k) = \boldsymbol{\mu}^c(k\tau) - \boldsymbol{\mu}_{tr}(k\tau)$ for $\boldsymbol{\mu} = \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{z}, \boldsymbol{y}$ and for all nonnegative integers k, with $\boldsymbol{z}_{tr} = \boldsymbol{g}(\boldsymbol{x}_{tr}, \boldsymbol{w}_{tr}, \boldsymbol{u}_{tr})$ and $\boldsymbol{y}_{tr} = \boldsymbol{h}(\boldsymbol{x}_{tr}, \boldsymbol{w}_{tr})$. The state-space matrices in the above system equation are given by

$$\begin{aligned}\boldsymbol{A} &= \boldsymbol{I} + \tau\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}^c}\bigg|_{tr},\ \boldsymbol{B}_1 = \tau\boldsymbol{W}_w\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{w}^c}\bigg|_{tr}\\ \boldsymbol{B}_2 &= \tau\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}^c}\bigg|_{tr},\ \boldsymbol{C}_1 = \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}^c}\bigg|_{tr},\ \boldsymbol{D}_{11} = \boldsymbol{W}_w\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{w}^c}\bigg|_{tr}\\ \boldsymbol{D}_{12} &= \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}^c}\bigg|_{tr},\ \boldsymbol{C}_2 = \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}^c}\bigg|_{tr},\ \boldsymbol{D}_{21} = \boldsymbol{W}_w\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{w}^c}\bigg|_{tr}.\end{aligned}$$

In the preceding expressions, $|_{tr}$ indicates that the matrix entries are evaluated at $(\boldsymbol{x}_{tr}, \boldsymbol{w}_{tr}, \boldsymbol{u}_{tr})$. $\boldsymbol{W}_w = \mathrm{diag}(\boldsymbol{I}_3, 0.5\boldsymbol{I}_3, 2, 0.01\boldsymbol{I}_3, \boldsymbol{I}_3, 0.01)$ is a weighting matrix used to scale the disturbance inputs, where $\mathrm{diag}(A_1, \ldots, A_n)$ denotes the block-diagonal augmentation of the matrices $A_1, \ldots, A_n$, and $\boldsymbol{I}_n$ denotes an $n \times n$ identity matrix.

A standard LTI $\mathcal{H}_\infty$ controller (with an ℓ_2-gain performance level of $\gamma = 1.68$) is designed for the discrete-time LTI system (1) following the procedure outlined in [28]. The semidefinite programs in the controller synthesis problem are solved in MATLAB using YALMIP [29] with MOSEK [30]. While this controller is specifically designed to force the aircraft to track a straight and level trim trajectory, we are interested in tracking trajectories that involve changes in the path curvature and altitude. Factoring in the maneuvering capabilities of the encroacher and the defenders, we have found that the designed controller still performs reasonably well for the ensuing trajectories. When implementing the controller, instead of using the reference state and control values corresponding to the straight and level trim trajectory, we use trim state and control input pertaining to the current path curvature and flight path angle of the reference trajectory. Namely, by solving the nonlinear system equations for different values of the path curvature and flight path angle, we parameterize the trim state and control input in terms of the path curvature and flight path angle. Then, as in [25], the variations of the trim state and control input with the path curvature and flight path angle are approximated by polynomial functions. During implementation, we deduce the current path curvature and flight path angle of the reference trajectory from the current and past positions of the virtual vehicle. At each time instant, the trim state and control input corresponding to the current path curvature and flight path angle are computed from the polynomial functions, and are then used to calculate $\bar{\boldsymbol{x}}$ and $\bar{\boldsymbol{u}}$. We have found that using the parameterized trim state and control input leads to improved tracking performance for trajectories of interest.

Next, we present the specifications for defender 1, its environment, and its higher-level property in TLA. In the sequel, a component's lower-level specification, higher-level property, and environmental assumptions are denoted by the symbols M_j, P_j, and E_j, respectively, where the subscript j is d_1 for defender 1 and d_2 for defender 2. $Init_{(\cdot)}$, $\Theta_{(\cdot)}$, and $v_{(\cdot)}$ in a TLA formula denote a state predicate, the tuple of internal variables, and the tuple of variables pertinent to a component's output, respectively.

D. SPECIFICATION OF DEFENDER 1 AND ITS ENVIRONMENT

Defender 1 performs the following tasks: (1) senses the position and velocity of the encroacher, (2) determines its mode of operation and computes the next position of the virtual vehicle using Algorithm 1, and (3) employs the discrete-time LTI controller described in Section IV-C to track the virtual vehicle despite exogenous disturbances. The inputs to defender 1 are the position and velocity of the encroacher and the exogenous disturbances. We now express the environmental assumptions of defender 1 in TLA.

1) ENVIRONMENTAL ASSUMPTIONS

First, we specify the assumptions on the encroacher's capabilities. Let P_e and V_e denote the position vector and velocity vector of the encroacher, respectively. As mentioned earlier, we make assumptions on the velocities, accelerations, and maneuvering capabilities of the encroacher. These assumptions are specified in TLA using the following formulas:

$$\begin{aligned}
\overline{Init}_{en} &\triangleq P_e \in \mathbb{R}^3 \wedge V_e \in \mathbb{R}^3 \wedge \|V_e\| \leq \mathtt{V_{e_{max}}} \\
&\quad \wedge \|V_e\| \geq \mathtt{V_{e_{min}}} \wedge |V_e(3)| \leq \mathtt{V_{ze_{max}}} \\
Init_{en} &\triangleq \overline{Init}_{en} \wedge P_e(3) \leq -60 \\
\mathcal{N}_{en} &\triangleq P'_e - P_e = \tau V_e \wedge (|V'_e - V_e| \leq \tau[7, 7, 3]^T) \\
&\quad \wedge R'_e = \text{getRCurv}(P_e, P'_e) \wedge |R'_e| \geq 80 \wedge \overline{Init}'_{en}.
\end{aligned}$$

$Init_{en}$ and $\mathcal{N}_{en}$ are the initial state predicate and the next state action, respectively. The function getRCurv takes the position of the encroacher at the current state and the next state as inputs and returns the radius of curvature R_e. The preceding formula specifies limits on the total speed, vertical speed, and accelerations of the encroacher. The limit on the minimum radius of curvature of the encroacher's trajectory is specified through the predicate $|R'_e| \geq 80$.

Next, we specify the assumptions on the exogenous disturbances that affect defender 1. The exogenous disturbances are represented by a 16-channel signal. The first two disturbance channels represent constant wind disturbances in the NE-plane, and they are specified as TLA rigid variables whose values respect some predefined bounds. Specifically, we assume that the resultant steady wind does not exceed $\mathtt{W_{st}}$, which in this case study equals 2.5 m/s. The next three disturbance channels represent white noise signals, which are fed into the Dryden turbulence model [31] (low altitude, moderate turbulence) to give the velocity perturbations due to atmospheric turbulence. The remaining eleven channels represent additive white noise in the UAS sensor measurements. In TLA, we specify a white noise signal using its autocorrelation [32]. Given a sequence of length N denoted by $\epsilon(\cdot)$ and generated from a discrete-time white noise signal, we have the following constraint: $\Sigma_{j=1}^{\mathtt{N}} \epsilon(\text{mod}(j+i, \mathtt{N}))\epsilon(j) = 0 \quad \forall\, i \in \{1, \ldots, \mathtt{N}\}$, where $\text{mod}(a, b)$ denotes the operation a modulo b. White noise can be represented using this constraint by allowing N to become arbitrarily large. The exogenous disturbances are expressed by the following predicate and action:

$$\begin{aligned}
Init_{dt_1} &\triangleq D_1 \in \mathbb{R}^{16\times \mathtt{N}} \wedge D_1^T = [d_1^T, d_2^T, \ldots, d_{16}^T] \\
&\quad \wedge t_e = 1 \wedge w_1 = D_1(:, t_e) \wedge \|[d_1(1), d_2(1)]\| \leq \mathtt{W_{st}} \\
&\quad \wedge (\forall i \in \{1, 2\},\ \forall j \in \{2, \ldots, \mathtt{N}\},\ d_i(j) = d_i(1)) \\
&\quad \wedge (\forall i \in \{3, \ldots, 16\},\ \|d_i\| = \sqrt{\mathtt{N}}) \\
&\quad \wedge (\forall i \in \{3, \ldots, 16\},\ \forall j \in \{1, \ldots, \mathtt{N}\}, \\
&\quad \Sigma_{k=1}^{\mathtt{N}}\, d_i(\text{mod}(k+j, \mathtt{N}))\, d_i(k) = 0) \\
\mathcal{N}_{dt_1} &\triangleq t_e < t_1 \wedge t_e < t_2 \wedge t'_e = t_e + 1 \wedge t_e < \mathtt{N} \\
&\quad \wedge w'_1 = D_1(:, t'_e) \wedge D'_1 = D_1
\end{aligned}$$

where w_1 is a TLA variable that represents the disturbance vector, t_e is a variable that increments by 1 whenever a $\mathcal{N}_{dt_1}$ step occurs. $t_e < t_1$ and $t_e < t_2$ are enabling conditions for $\mathcal{N}_{dt_1}$, where t_1 and t_2 are variables specified in the lower-level

specifications of defender 1 and defender 2, respectively. The purpose of the enabling conditions is to ensure that an $\mathcal{N}_{dt_1}$ step is always followed by a step for each of defender 1 and defender 2.

Having defined the necessary predicates and actions, the complete specification of defender 1's environment can be written as the following TLA formula:

$$E_{d_1} \triangleq Init_{e_{d_1}} \wedge \Box[\mathcal{N}_{e_{d_1}}]_{\langle \Theta_{e_{d_1}}, v_{e_{d_1}} \rangle}, \text{ where}$$

$$Init_{e_{d_1}} \triangleq Init_{en} \wedge Init_{dt_1}$$

$$\mathcal{N}_{e_{d_1}} \triangleq (\mathcal{N}_{en} \wedge \mathcal{N}_{dt_1}) \vee Unch(\langle P_e, V_e, t_e, w_1 \rangle)$$

$$\Theta_{e_{d_1}} \triangleq \langle R_e, D_1, d_1, \ldots, d_{16} \rangle$$

$$v_{e_{d_1}} \triangleq \langle P_e, V_e, t_e, w_1 \rangle.$$

2) LOWER-LEVEL SPECIFICATIONS

We now express how defender 1 responds to its environmental inputs and generates the pertinent outputs. The evolution of the physical system is governed by a set of nonlinear differential equations with the UAS control input vector and the disturbance vector w_1 as inputs. The control input vector is generated by the discrete-time LTI $\mathcal{H}_\infty$ controller described in Section IV-C. The equations of motion presented in Section III-B pertain to the nominal system without any uncertainties. As a result of the uncertainties that arise from various sources, there are bound to be discrepancies between the UAS model and the physical system. Therefore, to better characterize the actual physical system, we incorporate uncertainties into the nominal UAS equations of motion. Specifically, we consider uncertainties in the aerodynamic coefficients and actuator models.

The following describes how the uncertainties are sampled during each step; the uncertainty characterization is adopted from [33]. The aerodynamic uncertainties, denoted by δC_i^1 for $i \in \{x, y, z, l, m, n\}$, are modeled as static rate-bounded perturbations with prescribed bounds on their magnitudes. We consider uncertainties for all four actuators, and each uncertainty is modeled as an uncertain dynamic LTI system whose transfer matrix has a bounded $\mathcal{H}_\infty$ norm. For $i \in \{E, A, R, T\}$, let $A_{\Delta_i^1}$, $B_{\Delta_i^1}$,$C_{\Delta_i^1}$, $D_{\Delta_i^1}$ denote the matrices pertaining to the state-space realization for the actuator dynamics uncertainty Δ_i^1, where E, A, R, and T denote the elevator, aileron, rudder, and throttle, respectively. In the following TLA specifications, for the sake of brevity, we will denote the sets $\{x, y, z, l, m, n\}$ and $\{E, A, R, T\}$ by A_a and A_c, respectively. The initialization and evolution of the uncertainties are specified as given below:

$$\begin{aligned} Init_{\Delta_1} \triangleq\ & (\forall i \in A_a,\ \delta C_i^1 \leq \overline{\delta \mathtt{C}_\mathtt{i}} \wedge \delta C_i^1 \geq \underline{\delta \mathtt{C}_\mathtt{i}}) \\ & \wedge (\forall i \in A_c,\ \text{SysReal}(A_{\Delta_i^1}, B_{\Delta_i^1}, C_{\Delta_i^1}, D_{\Delta_i^1}, \bar{\Delta}^{\mathtt{i}})) \\ & \wedge \delta C^1 = \langle \delta C_x^1, \ldots, \delta C_n^1 \rangle \wedge \delta A^1 = \langle A_{\Delta_E^1}, B_{\Delta_E^1}, \\ & \quad C_{\Delta_E^1}, D_{\Delta_E^1}, \ldots, A_{\Delta_T^1}, B_{\Delta_T^1}, C_{\Delta_T^1}, D_{\Delta_T^1} \rangle \end{aligned}$$

$$\begin{aligned} \mathcal{N}_{\Delta_1} \triangleq\ & Unch(\delta A^1) \wedge (\forall i \in A_a,\ (\delta C_i^1)' \in [\underline{\delta \mathtt{C}_\mathtt{i}}, \overline{\delta \mathtt{C}_\mathtt{i}}] \\ & \wedge (\delta C_i^1)' - \delta C_i^1 \in [\underline{\delta \mathtt{C}_\mathtt{i}}^{\mathtt{rb}}, \overline{\delta \mathtt{C}_\mathtt{i}}^{\mathtt{rb}}]) \\ & \wedge (\delta C^1)' = \langle (\delta C_x^1)', \ldots, (\delta C_n^1)' \rangle \end{aligned}$$

where $\text{SysReal}(A_{\Delta_i^1}, B_{\Delta_i^1}, C_{\Delta_i^1}, D_{\Delta_i^1}, \bar{\Delta}^{\mathtt{i}})$ prescribes the state-space matrices for Δ_i^1 such that the transfer matrix is well-defined and has an $\mathcal{H}_\infty$ norm less than $\bar{\Delta}^{\mathtt{i}}$. The values of the constants $\underline{\delta \mathtt{C}_\mathtt{i}}$, $\overline{\delta \mathtt{C}_\mathtt{i}}$, $\underline{\delta \mathtt{C}_\mathtt{i}}^{\mathtt{rb}}$, $\overline{\delta \mathtt{C}_\mathtt{i}}^{\mathtt{rb}}$, and $\bar{\Delta}^{\mathtt{j}}$ for $i \in A_a$ and $j \in A_c$ are the same as those given in [33].

Next, we specify the dynamics of the open-loop UAS in the following predicate and action:

$$\begin{aligned} Init_{ol_1} \triangleq\ & x_1 \in \mathbb{R}^{n_{x_1}} \wedge x_1(1{:}8) = \mathtt{x_{tr}}(1{:}8) \\ & \wedge x_1(9) \in [0, 2\pi) \wedge x_1(13{:}n_{x_1}) = 0 \\ & \wedge \|P_e(1{:}2) - x_1(10{:}11)\| \in [\mathtt{D_3} - \mathtt{D_s}/2, \mathtt{D_3} + \mathtt{D_s}/2] \\ & \wedge \|x_1(10{:}11) - x_2(10{:}11)\| \in [0, \mathtt{D_s}] \\ & \wedge P_e(3) - x_1(12) \in [(\mathtt{H_{of}} - \mathtt{D_s})/2, (\mathtt{H_{of}} + \mathtt{D_s})/2] \\ & \wedge x_2(12) - x_1(12) \in [\mathtt{H_{of}} - \mathtt{D_s}, \mathtt{H_{of}} + \mathtt{D_s}] \\ \mathcal{N}_{ol_1} \triangleq\ & x_1' = \text{Integrate}(F(x_1, u_1, w_1, \delta C^1, \delta A^1), \tau) \end{aligned}$$

where x_1 and u_1 are the state and control input vectors of defender 1, respectively, and $\mathtt{x_{tr}} = \boldsymbol{x}_{tr}(0)$. x_2 in the above formula is the state vector of defender 2. The size of the state vector, denoted by n_{x_1}, varies depending on the number of state variables of the actuator dynamics uncertainties. The formula $Init_{ol_1}$ specifies the initial position of defender 1 based on the initial configuration of the multi-aircraft system as described earlier. The formula $\text{Integrate}(F(x_1, u_1, w_1, \delta C^1, \delta A^1), \tau)$ performs integration of $F(x_1, u_1, w_1, \delta C^1, \delta A^1)$, which represents the UAS nonlinear system of differential equations with the previously described uncertainties, over a period of τ s starting from the state x_1 and holding the inputs u_1 and w_1 constant during the period of integration. The TLA specification of the integral operation is very similar to the one provided in [34] and is not provided here; see [34] for further details. We now specify the controller that uses the position of the virtual vehicle, P_{v_1}, and the sensor measurements to compute the control input vector u_1 as given below:

$$\begin{aligned} Init_{ct_1} \triangleq\ & u_1 = \mathtt{u_{tr}} \wedge x_{k_1} = \mathbf{0}_{18\times 1} \wedge \kappa_1 = 0 \wedge \eta_1 = 0 \\ \mathcal{N}_{ct_1} \triangleq\ & \eta_1' = \text{getfpa}(P_{v_1}, P_{v_1}') \\ & \wedge \kappa_1' = \text{getCurv}(P_{v_1}, P_{v_1}') \wedge x_{k_1}' = \mathtt{A_{k_1}} x_{k_1} + \\ & \quad \mathtt{B_{k_1}}(\text{Sens}(x_1, w_1) - \text{RefOp}(P_{v_1}, \kappa_1, \eta_1)) \\ & \wedge u_1' = \mathtt{C_{k_1}} x_{k_1} + \mathtt{D_{k_1}}(\text{Sens}(x_1, w_1) - \\ & \text{RefOp}(P_{v_1}, \kappa_1, \eta_1)) + \text{uTrim}(\kappa_1, \eta_1) \end{aligned}$$

where $\mathtt{u_{tr}} = \boldsymbol{u}_{tr}(0)$. The function Sens takes the UAS state vector and the disturbance vector as inputs and returns the sensor measurements. The functions getCurv and getfpa take the current and the past virtual vehicle positions as inputs and return the current curvature and the flight path angle of the virtual vehicle trajectory, respectively. The function uTrim takes the curvature and flight path angle of the reference

trajectory as inputs and returns the trim control input. The function RefOp takes the position of the virtual vehicle and the curvature and flight path angle of the trajectory as inputs and returns the reference output. The matrices $\mathtt{A}_{k_1}, \mathtt{B}_{k_1}, \mathtt{C}_{k_1}$, and $\mathtt{D}_{k_1}$ denote the state-space matrices pertaining to the LTI $\mathcal{H}_\infty$ controller described in Section IV-C. To finish specifying defender 1, we provide formulas for defender 1's perception of the encroacher's position and velocity, the computation of its mode of operation, and the evolution of its virtual vehicle as given below:

$$
\begin{aligned}
Init_{sn_1} \triangleq\ & P_{e_1} = P_e \wedge V_{e_1} = V_e \wedge P_1 = x_1(10{:}12) \\
& \wedge t_1 = 1 \wedge T_{max} \in \mathbb{R}_+ \wedge T_{max} \leq 151 \wedge t_1 \leq T_{max} \\
& \wedge \theta_{de} = \tan^{-1}\left((P_1(1) - P_{e_1}(1))/(P_1(2) - P_{e_1}(2))\right) \\
& \wedge P_{v_1}(1{:}2) = P_{e_1}(1{:}2) + \mathtt{D}_3[\sin\theta_{de},\ \cos\theta_{de}]^T \\
& \wedge P_{v_1}(3) = P_{e_1}(3) - \mathtt{H}_{\mathtt{of}}/2 \wedge \hat{P}_{v_2}(1{:}2) = P_{v_1}(1{:}2) \\
& \wedge \hat{P}_{v_2}(3) = P_{v_1}(3) + \mathtt{H}_{\mathtt{of}} \wedge \mathrm{Mode}_{\mathrm{swl}} \\
\mathrm{Mode}_{\mathrm{swl}} \triangleq\ & (\mathrm{Mode}_1 = 1 \wedge R_1 \wedge V_{e_1}(3) \leq 0) \\
& \vee (\mathrm{Mode}_1 = 1 \wedge |P_{v_1}(3) - P_{e_1}(3)| \leq \mathtt{H}_{\mathtt{of}}/2 \wedge \neg R_1) \\
& \vee (\mathrm{Mode}_1 = 1 \wedge R_2 \wedge V_{e_1}(3) \leq 0) \\
& \vee (\mathrm{Mode}_1 = 2 \wedge |P_{v_1}(3) - P_{e_1}(3)| > \mathtt{H}_{\mathtt{of}}/2 \wedge \neg R_2) \\
& \vee (\mathrm{Mode}_1 = 2 \wedge R_1 \wedge V_{e_1}(3) > 0) \\
& \vee (\mathrm{Mode}_1 = 2 \wedge R_2 \wedge V_{e_1}(3) > 0) \\
\mathcal{N}_{sn_1} \triangleq\ & t_1 \leq T_{max} \wedge t_1 \leq t_e \wedge t_1 = t_2 \wedge P'_{e_1} = P_e \\
& \wedge V'_{e_1} = V_e \wedge t'_1 = t_1 + 1 \wedge Unch(T_{max}) \\
& \wedge \langle P'_{v_1}, \hat{P}'_{v_2} \rangle = \mathrm{updateVV}(P_{v_1}, \hat{P}_{v_2}, P_{e_1}, V_{e_1}, \\
& \qquad \mathrm{Mode}_1) \wedge P'_1 = x'_1(10{:}12) \wedge \mathrm{Mode}'_{\mathrm{swl}}
\end{aligned}
$$

where the predicates R_1 and R_2 are given as follows:

$$
\begin{aligned}
R_1 \triangleq\ & \hat{P}_{v_2}(3) - P_{e_1}(3) \geq \mathtt{H}_{\mathtt{of}}/2 \\
& \wedge \|P_{v_1} - P_{e_1}\| \in [\mathtt{D}_1 - \epsilon_1, \mathtt{D}_1] \\
& \wedge \|\hat{P}_{v_2} - P_{e_1}\| \in [\mathtt{D}_1 - \epsilon_1, \mathtt{D}_1] \\
R_2 \triangleq\ & P_{e_1}(3) - P_{v_1}(3) > \mathtt{H}_{\mathtt{of}}/2 \\
& \wedge \|P_{v_1} - P_{e_1}\| \in [\mathtt{D}_1 - \epsilon_1, \mathtt{D}_1] \\
& \wedge \|\hat{P}_{v_2} - P_{e_1}\| \in [\mathtt{D}_1 - \epsilon_1, \mathtt{D}_1].
\end{aligned}
$$

T_{max} in the preceding formulas is less than or equal to 151 and is computed based on the total mission duration and the sampling time. $\mathrm{Mode}_{\mathrm{swl}}$ is a predicate that specifies the logic used to determine the mode of operation of defender 1. The schematic diagram shown in Fig. 2 summarizes this logic. As seen from the figure, the mode of operation is different in the regions defined by the predicates R_1 and R_2 depending on whether the encroacher is going up or down. The reason for this difference is to ensure that when the mode of operation switches between 1 and 2, the encroacher would still be within a distance of $\mathtt{D}_1$ from one of the defenders. P_1, $\hat{P}_{v_2}$, and P_{e_1} in the above specification denote the position of defender 1, the position of defender 2's virtual vehicle as estimated by defender 1, and the position of the encroacher as sensed by defender 1, respectively. $\mathrm{updateVV}(\cdot,\cdot,\cdot,\cdot,\cdot)$ is a function that takes as inputs the current position of defender 1's virtual vehicle, the estimated current position of defender 2's virtual vehicle, the current position and velocity of the encroacher, and the mode of operation and returns the updated position of defender 1's virtual vehicle and the estimated position of defender 2's virtual vehicle at the next time instant using Algorithm 1. There is a one step lag between the actual position of the encroacher and its position as sensed by the defender. This one step lag is not only a physical necessity in order to take into account delays due to perception, it is also essential to ensure that the hypothesis (2*a*) of Theorem 1 holds, thereby enabling us to verify the global system property.

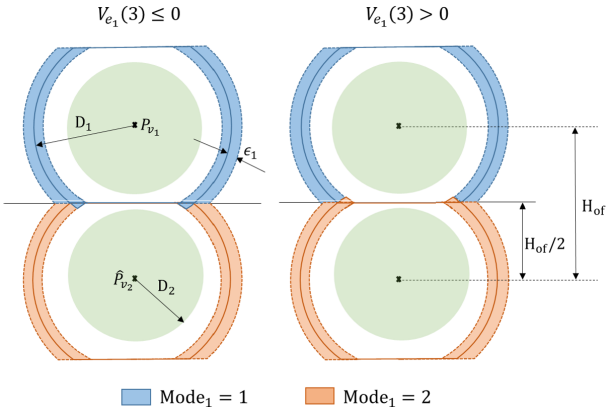

FIGURE 2. Schematic showing the virtual vehicle positions of the two defenders along with the relevant dimensions (some of the dimensions are exaggerated for clarity).

From $Init_{ol_1}$ and $Init_{sn_1}$, we observe that the initial planar positions of defender 1, its virtual vehicle, and the encroacher are collinear. Noting that the maximum planar distance between defender 1 and its virtual vehicle is less than or equal to $\mathtt{D}_s/2$, we deduce the following:

$$
Init_{ol_1} \wedge Init_{sn_1} \Rightarrow \|P_{v_1} - P_1\| \leq \mathtt{D}_2/2. \tag{2}
$$

The above inequality asserts that during the start of the mission, the distance between defender 1 and its virtual vehicle is at most $\mathtt{D}_2/2$. This information will be used later in this section to verify (5).

We now give the lower-level specification for defender 1 as the following TLA formula:

$$
\begin{aligned}
M_{d_1} \triangleq\ & \exists \Theta_{d_1} : Init_{d_1} \wedge \Box[\mathcal{N}_{d_1}]_{\langle \Theta_{d_1}, v_{d_1} \rangle} \\
& \wedge WF_{\langle \Theta_{d_1}, v_{d_1} \rangle}(\mathcal{N}_{d_1}), \text{ where} \\
Init_{d_1} \triangleq\ & Init_{\Delta_1} \wedge Init_{ol_1} \wedge Init_{ct_1} \wedge Init_{sn_1} \\
\mathcal{N}_{d_1} \triangleq\ & \mathcal{N}_{\Delta_1} \wedge \mathcal{N}_{ol_1} \wedge \mathcal{N}_{ct_1} \wedge \mathcal{N}_{sn_1} \\
\Theta_{d_1} \triangleq\ & \langle \delta C^1, \delta A^1, x_1, \kappa_1, \eta_1, u_1, x_{k_1}, V_{e_1}, \mathrm{Mode}_1, T_{max} \rangle \\
v_{d_1} \triangleq\ & \langle P_{v_1}, \hat{P}_{v_2}, P_{e_1}, P_1, t_1 \rangle.
\end{aligned}
$$

For defender 1, our task is to demonstrate that the conjunction of the environmental and lower-level process specifications, E_{d_1} and M_{d_1}, implies a higher-level property, which

is a desirable attribute that we wish to formally verify. This property amounts to providing a bound on the aircraft's position error from the virtual vehicle and guaranteeing that the virtual vehicle of defender 1 is always at a distance of $\mathtt{D}_1$ from the encroacher when $\text{Mode}_1{=}1$ and the distance between the two virtual vehicles is always $\mathtt{H}_{\mathtt{of}}$. Such a property can be expressed by the following TLA formula:

$$\begin{aligned}
P_{d_1} &\triangleq Init_{p_{d_1}} \wedge \Box[\mathcal{N}_{p_{d_1}}]_{\langle v_{d_1}\rangle}, \text{ where}\\
\mathcal{N}_{p_{d_1}} &\triangleq \mathcal{N}^1_{p_{d_1}} \wedge \mathcal{N}^2_{p_{d_1}}\\
Init_{p_{d_1}} &\triangleq t_1 = 1 \wedge \mathcal{N}_{p_{d_1}}\\
\mathcal{N}^1_{p_{d_1}} &\triangleq \{\|P_{v_1} - P_{e_1}\| = \mathtt{D}_1 \wedge \|\hat{P}_{v_2} - P_{e_1}\| \geq \mathtt{D}_1 - \epsilon_1\\
&\quad \wedge \|\hat{P}_{v_2} - P_{v_1}\| = \mathtt{H}_{\mathtt{of}}\} \vee \{\|\hat{P}_{v_2} - P_{e_1}\| = \mathtt{D}_1\\
&\quad \wedge \|P_{v_1} - P_{e_1}\| \geq \mathtt{D}_1 - \epsilon_1 \wedge \|\hat{P}_{v_2} - P_{v_1}\| = \mathtt{H}_{\mathtt{of}}\}\\
\mathcal{N}^2_{p_{d_1}} &\triangleq t_1 \leq T_{max} \wedge \|P_1 - P_{v_1}\| \leq \mathtt{D}_2\,.
\end{aligned}$$

The following lemma states an intermediate result that will be used in Section IV-F to verify the global system property.

Lemma 2: Given the constants and system design parameters as in Table 2, the following holds:

$$Init_{d_1} \wedge \Box[\mathcal{N}_{d_1}]_{\langle \Theta_{d_1}, v_{d_1}\rangle} \Rightarrow \Box[\mathcal{N}^1_{p_{d_1}}]\,.$$

Proof: We will use the INV1 proof rule from [22], which is provided below, to prove the lemma:

$$\frac{I \wedge [\mathcal{N}]_f \Rightarrow I'}{I \wedge \Box[\mathcal{N}]_f \Rightarrow \Box[I]}\,.$$

The proof follows from verifying

$$Init_{d_1} \Rightarrow \mathcal{N}^1_{p_{d_1}} \tag{3}$$

$$\text{and } \mathcal{N}^1_{p_{d_1}} \wedge [\mathcal{N}_{d_1}]_{\langle \Theta_{d_1}, v_{d_1}\rangle} \Rightarrow (\mathcal{N}^1_{p_{d_1}})'\,. \tag{4}$$

Since verifying (3) is straightforward, we focus only on (4). We observe that $\text{Mode}_{\text{swl}} \wedge [\mathcal{N}_{d_1}]_{\langle \Theta_{d_1}, v_{d_1}\rangle} \Rightarrow \text{Mode}'_{\text{swl}}$, and using INV1 we can conclude that Mode_{swl} is an invariant of $[\mathcal{N}_{d_1}]_{\langle \Theta_{d_1}, v_{d_1}\rangle}$. Therefore, using the proof rule INV2 of [22], $[\mathcal{N}_{d_1}]_{\langle \Theta_{d_1}, v_{d_1}\rangle}$ in (4) can be strengthened to $[\mathcal{N}_{d_1} \wedge \text{Mode}_{\text{swl}} \wedge \text{Mode}'_{\text{swl}}]_{\langle \Theta_{d_1}, v_{d_1}\rangle}$. Then, on expanding (4), we obtain twelve logical implications that need to be verified. For instance, one of these implications is given by

$$\begin{aligned}
&\mathcal{N}^{11}_{p_{d_1}} \wedge (\text{Mode}_1 = 1 \wedge |P_{v_1}(3) - P_{e_1}(3)| \leq \mathtt{H}_{\mathtt{of}}/2 \wedge \neg R_1)\\
&\wedge \langle P'_{v_1}, \hat{P}'_{v_2}\rangle = \text{updateVV}(P_{v_1}, \hat{P}_{v_2}, P_{e_1}, V_{e_1}, \text{Mode}_1)\\
&\wedge P'_{e_1} = P_e \Rightarrow (\mathcal{N}^{11}_{p_{d_1}})', \text{ where}\\
&\mathcal{N}^{11}_{p_{d_1}} \triangleq \|P_{v_1} - P_{e_1}\| = \mathtt{D}_1 \wedge \|\hat{P}_{v_2} - P_{e_1}\| \geq \mathtt{D}_1 - \epsilon_1\\
&\wedge \|\hat{P}_{v_2} - P_{v_1}\| = \mathtt{H}_{\mathtt{of}}\,.
\end{aligned}$$

This implication follows directly from Lemma 1. Note that the limit $\mathtt{H}_{\mathtt{of}}/2$ in the conditions $|dz(k)| \leq \mathtt{H}_{\mathtt{of}}/2$ and $|\hat{dz}(k)| \leq \mathtt{H}_{\mathtt{of}}/2$ that appear in the statement of Lemma 1 can be slightly increased to 16.2, and the results of the lemma can still be proven to be correct following the same proof arguments. With this said, the other eleven implications can also be shown to hold using Lemma 1 or the aforementioned slightly modified version of this lemma. The proof is then completed by invoking the INV1 proof rule. ■

Verifying the remaining term in the component property, namely $\mathcal{N}^2_{p_{d_1}}$, solely in the realm of formal methods is a formidable task due to the continuous nature of the state space. That said, tools from robust control such as IQC theory [21] can potentially be used to prove the following assertion:

$$E_{d_1} \wedge M_{d_1} \Rightarrow (t_1 = 1 \wedge \mathcal{N}^2_{p_{d_1}}) \wedge \Box[\mathcal{N}^2_{p_{d_1}}]_{\langle v_{d_1}\rangle}. \tag{5}$$

Work is currently underway in our group to develop results that would enable us to prove such an assertion. However, for the purpose of this paper, we will make use of Monte Carlo simulations to verify (5). Instead of using simulations to verify the assertion $E_{d_1} \wedge M_{d_1} \Rightarrow P_{d_1}$, we only use simulations to verify the simpler assertion (5), as we have already proven part of the component property in Lemma 2. This simplification is possible because of the modular architecture of the defender. Moreover, verifying the simpler property in (5) leads to fewer simulations by eliminating the need to simulate the motions of the encroacher and defender 2.

Monte Carlo simulations are performed by generating different encroacher trajectories that satisfy the constraints specified in $\mathcal{N}_{en}$. Since the encroacher is trying to evade the defenders, it is reasonable to assume that it is accelerating/decelerating as much as allowable by its propulsion system and control authority. In doing so, the encroacher's motion also needs to satisfy the velocity and radius of curvature constraints. We generate 100 different encroacher trajectories, each with a duration of 6 s. The procedure used to generate these trajectories is as follows. First, the three acceleration profiles of the encroacher trajectory are generated such that the accelerations may only change at $t = 2$ s and $t = 4$ s and at all other times the accelerations remain constant, where only the minimum and maximum values of the accelerations are used. Next, the resulting velocity profiles and trajectory are computed and then checked to determine whether they satisfy all the constraints specified in $\mathcal{N}_{en}$; if they do not, then the magnitudes of the accelerations are reduced by 5% of their current values and the procedure is repeated until a trajectory that satisfies all the constraints is obtained. This method of generating the encroacher trajectories allows for a wide range of maneuvers and not just ones with constant accelerations throughout the mission.

For each of the 100 trajectories, we perform 1000 simulations by sampling the uncertainties and varying the steady wind and sensor noise. The initial configurations of the defender and its virtual vehicle are also sampled according to the specifications $Init_{ol_1}$ and $Init_{sn_1}$, thereby satisfying (2). The simulation environment subjects the UAS to uncertainties and disturbances as specified in $\mathcal{N}_{\Delta_1}$ and $\mathcal{N}_{dt_1}$. In the simulations, the uncertainties and disturbances are pseudorandomly generated following a process similar to the one detailed in [33]. The simulation results are summarized in Fig. 3, where the top figure shows the distribution of the

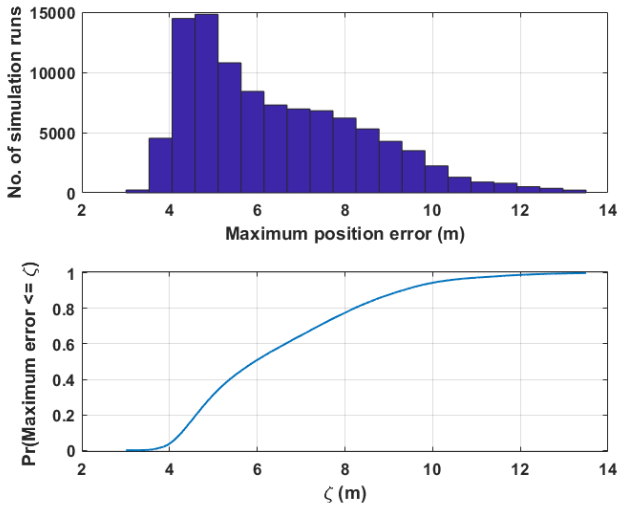

FIGURE 3. Results from Monte Carlo simulations; the top figure shows the distribution of the maximum position error in a simulation run, and the bottom figure depicts the corresponding cumulative distribution function.

maximum position error in a simulation run and the bottom figure illustrates the corresponding cumulative distribution function. From Fig. 3, we infer that the position error between defender 1 and its virtual vehicle is always less than 14 m, or in TLA notation, $\Box[t_1 \leq T_{max} \wedge \|P_1 - P_{v_1}\| \leq \mathtt{D}_2]$, hence verifying the implication in (5). Although the encroacher trajectories were designed to potentially include worst-case maneuvers and the simulation environment was set up to be a close representation of the physical processes, the observation that the simulation results satisfy (5) is by no means a proof of this implication. Our inability to mathematically show that (5) holds prevents us from providing a complete formal proof of the global system property despite the fact that all other component properties are formally proven. However, as argued in [4], guaranteeing absolute reliability of such complex systems is a formidable task; instead, our efforts should be focused on increasing reliability, and this is exactly what we are trying to achieve in this paper by adopting the modular design approach and the compositional verification framework. As all component properties are formally proven except for the implication in (5), if a violation of the global system property occurs even though all considered assumptions hold, then this violation can be attributed to the invalidity of (5), hence easily pinpointing the source of failure in post-analysis. With this said, the proof of the global system property provided in Section IV-G will take provisionally (5) to be correct.

E. SPECIFICATION OF DEFENDER 2 AND ITS ENVIRONMENT

The environmental assumptions and the component specifications for defender 2 can be expressed in a manner similar to that of defender 1. Since the specifications are very similar, we only provide the TLA formulas for the environmental assumptions, lower-level specifications, and the component property. The environmental assumptions of defender 2 are specified in the following TLA formula:

$$\begin{aligned}
E_{d_2} &\triangleq Init_{e_{d_2}} \wedge \Box[\mathcal{N}_{e_{d_2}}]_{\langle \Theta_{e_{d_2}}, v_{e_{d_2}} \rangle}, \text{ where} \\
Init_{e_{d_2}} &\triangleq Init_{en} \wedge Init_{dt_2} \\
\mathcal{N}_{e_{d_2}} &\triangleq (\mathcal{N}_{en} \wedge \mathcal{N}_{dt_2}) \vee Unch(\langle P_e, V_e, t_e, w_2 \rangle) \\
\Theta_{e_{d_2}} &\triangleq \langle R_e, D_2, \overline{d}_1, \ldots, \overline{d}_{16} \rangle \\
v_{e_{d_2}} &\triangleq \langle P_e, V_e, t_e, w_2 \rangle.
\end{aligned}$$

Next, we give the lower-level specification as follows:

$$\begin{aligned}
M_{d_2} &\triangleq \exists \Theta_{d_2} : Init_{d_2} \wedge \Box[\mathcal{N}_{d_2}]_{\langle \Theta_{d_2}, v_{d_2} \rangle} \\
&\quad \wedge WF_{\langle \Theta_{d_2}, v_{d_2} \rangle}(\mathcal{N}_{d_2}), \text{ where} \\
Init_{d_2} &\triangleq Init_{\Delta_2} \wedge Init_{ol_2} \wedge Init_{ct_2} \wedge Init_{sn_2} \\
\mathcal{N}_{d_2} &\triangleq \mathcal{N}_{\Delta_2} \wedge \mathcal{N}_{ol_2} \wedge \mathcal{N}_{ct_2} \wedge \mathcal{N}_{sn_2} \\
\Theta_{d_2} &\triangleq \langle \delta C^2, \delta A^2, x_2, \kappa_2, \eta_2, u_2, x_{k_2}, V_{e_2}, \text{Mode}_2 \rangle \\
v_{d_2} &\triangleq \langle P_{v_2}, \hat{P}_{v_1}, P_{e_2}, P_2, t_2 \rangle.
\end{aligned}$$

The higher-level property for defender 2 asserts the following: the position error between defender 2 and its virtual vehicle is always less than or equal to $\mathtt{D}_2$, its virtual vehicle is always at a distance of $\mathtt{D}_1$ from the encroacher when $\text{Mode}_2 = 1$, and the distance between the two virtual vehicles is always equal to $\mathtt{H}_{\mathtt{of}}$. This property is specified below:

$$\begin{aligned}
P_{d_2} &\triangleq Init_{p_{d_2}} \wedge \Box[\mathcal{N}_{p_{d_2}}]_{\langle v_{d_2} \rangle}, \text{ where} \\
\mathcal{N}_{p_{d_2}} &\triangleq \mathcal{N}^1_{p_{d_2}} \wedge \mathcal{N}^2_{p_{d_2}} \\
Init_{p_{d_2}} &\triangleq t_2 = 1 \wedge \mathcal{N}_{p_{d_2}} \\
\mathcal{N}^1_{p_{d_2}} &\triangleq \{\|P_{v_2} - P_{e_2}\| = \mathtt{D}_1 \wedge \|\hat{P}_{v_1} - P_{e_2}\| \geq \mathtt{D}_1 - \epsilon_1 \\
&\quad \wedge \|P_{v_2} - \hat{P}_{v_1}\| = \mathtt{H}_{\mathtt{of}}\} \vee \{\|\hat{P}_{v_1} - P_{e_2}\| = \mathtt{D}_1 \\
&\quad \wedge \|P_{v_2} - P_{e_2}\| \geq \mathtt{D}_1 - \epsilon_1 \wedge \|P_{v_2} - \hat{P}_{v_1}\| = \mathtt{H}_{\mathtt{of}}\} \\
\mathcal{N}^2_{p_{d_2}} &\triangleq t_2 \leq T_{max} \wedge \|P_2 - P_{v_2}\| \leq \mathtt{D}_2 .
\end{aligned}$$

The following result for defender 2 is analogous to Lemma 2.

Lemma 3: Given the constants and system design parameters as in Table 2, the following holds:

$$Init_{d_2} \wedge \Box[\mathcal{N}_{d_2}]_{\langle \Theta_{d_2}, v_{d_2} \rangle} \Rightarrow \Box[\mathcal{N}^1_{p_{d_2}}] .$$

The proof follows a similar argument to that used in the proof of Lemma 2 and is therefore omitted.

As with implication (2), we can deduce the following for defender 2:

$$Init_{ol_2} \wedge Init_{sn_2} \Rightarrow \|P_2 - P_{v_2}\| \leq \mathtt{D}_2/2. \tag{6}$$

We observe from $Init_{ol_2}$ and $Init_{sn_2}$ that $\|P_1(1{:}2) - P_{v_1}(1{:}2)\| \leq \mathtt{D}_s/2$ and $\|P_1(1{:}2) - P_2(1{:}2)\| \leq \mathtt{D}_s$, which result in $\|P_2(1{:}2) - P_{v_1}(1{:}2)\| \leq 3\mathtt{D}_s/2$. Since the planar positions of the two virtual vehicles coincide, we can write $\|P_2(1{:}2) - P_{v_2}(1{:}2)\| \leq 3\mathtt{D}_s/2$. Combining this with $\|P_2(3) - P_{v_2}(3)\| \leq \mathtt{D}_s/2$, we conclude that $\|P_2 - P_{v_2}\| \leq \mathtt{D}_2/2$.

Since defender 2 is governed by the same dynamics as defender 1, the assertion $E_{d_2} \wedge M_{d_2} \Rightarrow (t_2 = 1 \wedge \mathcal{N}^2_{p_{d_2}}) \wedge \Box[\mathcal{N}^2_{p_{d_2}}]_{\langle v_{d_2}\rangle}$ can also be verified using the Monte Carlo simulations presented in the previous section, along with (6). But, as mentioned before, regardless of how representative and extensive the simulations are, the results from these simulations will not constitute a definitive proof. Nonetheless, as in the case of defender 1, the problematic implication, i.e., the counterpart of (5), will be taken to be provisionally correct in the proof of the global system property.

F. GLOBAL SYSTEM AND ITS ENVIRONMENT

Now that we have formally defined the two components, the next step is to specify in TLA the global system property that we would like to verify. First, we specify the environment of the open system resulting from the conjoined specifications M_{d_1} and M_{d_2}, namely,

$$\begin{aligned}
E_{gl} &\triangleq Init_{e_{gl}} \wedge \Box[\mathcal{N}_{e_{gl}}]_{\langle \theta_{e_{gl}}, v_{e_{gl}}\rangle}, \text{ where}\\
Init_{e_{gl}} &\triangleq Init_{en} \wedge Init_{dt_1} \wedge Init_{dt_2}\\
\mathcal{N}_{e_{gl}} &\triangleq (\mathcal{N}_{en} \wedge \mathcal{N}_{dt_1} \wedge \mathcal{N}_{dt_2}) \vee Unch(\langle P_e, V_e, t_e, w_1, w_2\rangle)\\
\Theta_{e_{gl}} &\triangleq \langle \Theta_{e_{d_1}}, \Theta_{e_{d_2}}\rangle, \text{ and } v_{e_{gl}} \triangleq \langle P_e, V_e, t_e, w_1, w_2\rangle.
\end{aligned}$$

The global system property is as follows: throughout the time interval $[0, \tau(T_{max}-1)]$, at least one defender is within a distance of $\mathtt{D_a}$ from the encroacher, and each defender is at least a distance of $\mathtt{D_s}$ from the encroacher and the other defender. This property is expressed in TLA as follows:

$$\begin{aligned}
P_{gl} &\triangleq Init_{gl} \wedge \Box[\mathcal{N}_{p_{gl}}]_{\langle v_{gl}\rangle}, \text{ where}\\
Init_{gl} &\triangleq t_1 = 1 \wedge t_2 = 1 \wedge \mathcal{N}_{p_{gl}}\\
\mathcal{N}_{p_{gl}} &\triangleq t_1 \le T_{max} \wedge t_2 \le T_{max} \wedge (\mathcal{N}^1_{p_{gl}} \vee \mathcal{N}^2_{p_{gl}})\\
\mathcal{N}^1_{p_{gl}} &\triangleq \|P_1 - P_{e_1}\| \le \mathtt{D_a} \wedge \|P_1 - P_{e_1}\| \ge \mathtt{D_s}\\
&\quad \wedge \|P_2 - P_{e_2}\| \ge \mathtt{D_s} \wedge \|P_1 - P_2\| \ge \mathtt{D_s}\\
\mathcal{N}^2_{p_{gl}} &\triangleq \|P_2 - P_{e_2}\| \le \mathtt{D_a} \wedge \|P_2 - P_{e_2}\| \ge \mathtt{D_s}\\
&\quad \wedge \|P_1 - P_{e_1}\| \ge \mathtt{D_s} \wedge \|P_1 - P_2\| \ge \mathtt{D_s}\\
v_{gl} &\triangleq \langle P_{e_1}, P_{e_2}, P_1, P_2, t_1, t_2\rangle
\end{aligned}$$

where P_1 and P_2 are the positions of defender 1 and defender 2, respectively, and, as mentioned before, P_{e_1} and P_{e_2} denote the positions of the encroacher as sensed by defender 1 and defender 2, respectively. The predicate $\mathcal{N}^1_{p_{gl}}$, for example, asserts the following: defender 1 is within a distance of $\mathtt{D_a}$ from the encroacher, defender 1 is at a minimum distance of $\mathtt{D_s}$ from the encroacher and defender 2, and defender 2 is at a minimum distance of $\mathtt{D_s}$ from the encroacher.

G. VERIFYING THE GLOBAL SYSTEM PROPERTY

Given the global system property, the environmental assumptions, and the component specifications and properties, we want to show the following implication: $\models E_{gl} \wedge M_{d_1} \wedge M_{d_2} \Rightarrow P_{gl}$. The proof is carried out in two steps. The first step involves showing $\models E_{gl} \wedge M_{d_1} \wedge M_{d_2} \Rightarrow P_{d_1} \wedge P_{d_2}$, and the second step entails proving $P_{d_1} \wedge P_{d_2} \Rightarrow P_{gl}$. The first step is accomplished through the following theorem, which is proved using Theorem 1.

Theorem 2: Consider the system shown in Fig. 1 with the specifications and environmental assumptions as given in Sections IV-D, IV-E, and IV-F. Suppose that implication (5) *for defender* 1 *and its counterpart for defender* 2 *are valid. Then, the following holds*:

$$\models E_{gl} \wedge M_{d_1} \wedge M_{d_2} \Rightarrow P_{d_1} \wedge P_{d_2}.$$

Proof: The proof involves showing that the hypotheses (1), (2a), (2b), and (3) of Theorem 1 hold for the two components. The result then follows from invoking Theorem 1.

1) PROOF OF HYPOTHESIS (1)

We first show that hypothesis (1) holds for defender 1 by proving $\models \mathcal{C}(E_{gl}) \wedge \mathcal{C}(P_{d_1}) \wedge \mathcal{C}(P_{d_2}) \Rightarrow E_{d_1}$. Recognizing that E_{gl}, P_{d_1}, and P_{d_2} are safety properties and using the fact that the closure of a safety property is itself [10], the preceding assertion reduces to

$$\begin{aligned}
\models\ & \big(Init_{e_{gl}} \wedge Init_{p_{d_1}} \wedge Init_{p_{d_2}}\big)\\
& \wedge \Box[(\mathcal{N}_{e_{gl}} \vee Unch(\langle \Theta_{e_{gl}}, v_{e_{gl}}\rangle))\\
& \wedge (\mathcal{N}_{p_{d_1}} \vee Unch(\langle v_{d_1}\rangle)) \wedge (\mathcal{N}_{p_{d_2}} \vee Unch(\langle v_{d_2}\rangle))]\\
& \Rightarrow Init_{e_{d_1}} \wedge \Box[\mathcal{N}_{e_{d_1}} \vee Unch(\langle \Theta_{e_{d_1}}, v_{e_{d_1}}\rangle)]. \qquad (7)
\end{aligned}$$

Substituting the specifications from the previous sections in (7) and employing step simulation, the above implication follows immediately. Using a similar argument, hypothesis (1) can also be shown to hold for defender 2.

2) PROOF OF HYPOTHESIS (2b) FOR DEFENDER 1

Since it is convenient to prove hypothesis (2b) before (2a), we first show that hypothesis (2b) holds for defender 1 by proving the following:

$$\models E_{d_1} \wedge M_{d_1} \Rightarrow P_{d_1}. \qquad (8)$$

Given that (5) is valid and that $\mathcal{N}^1_{p_{d_1}}$ is an invariant of $[\mathcal{N}_{d_1}]_{\langle \Theta_{d_1}, v_{d_1}\rangle}$ based on Lemma 2, then using the proof rule INV2 of [22] we can strengthen $Init_{d_1}$ and $[\mathcal{N}_{d_1}]_{\langle \Theta_{d_1}, v_{d_1}\rangle}$ in (5) to $Init_{d_1} \wedge \mathcal{N}^1_{p_{d_1}}$ and $[\mathcal{N}_{d_1} \wedge \mathcal{N}^1_{p_{d_1}} \wedge (\mathcal{N}^1_{p_{d_1}})']_{\langle \Theta_{d_1}, v_{d_1}\rangle}$, respectively, thereby showing that (8) holds.

3) PROOF OF HYPOTHESIS (2a) FOR DEFENDER 1

We now show that hypothesis (2a) holds for defender 1 by showing the following:

$$\models \mathcal{C}(E_{d_1})_{+v_{d_1}} \wedge \mathcal{C}(M_{d_1}) \Rightarrow \mathcal{C}(P_{d_1}). \qquad (9)$$

First, since $\mathcal{C}(E_{d_1})_{+v_{d_1}} = \mathcal{C}((E_{d_1})_{+v_{d_1}})$, (9) can be equivalently written as

$$\models \mathcal{C}((E_{d_1})_{+v_{d_1}}) \wedge \mathcal{C}(M_{d_1}) \Rightarrow \mathcal{C}(P_{d_1}). \qquad (10)$$

By virtue of Proposition 3 of [10], $(E_{d_1})_{+v_{d_1}}$ is expressed as

$$(E_{d_1})_{+v_{d_1}} \equiv \exists s : \ \overline{Init}_{e_{d_1}} \wedge \Box[\overline{\mathcal{N}}_{e_{d_1}}]_{\bar{v}_{e_{d_1}}}$$

$$
\begin{aligned}
\overline{Init}_{e_{d_1}} &\triangleq (Init_{e_{d_1}} \wedge (s=0)) \vee (\neg Init_{e_{d_1}} \wedge (s=1)) \\
\overline{\mathcal{N}}_{e_{d_1}} &\triangleq \overline{\mathcal{N}}^1_{e_{d_1}} \vee \overline{\mathcal{N}}^2_{e_{d_1}} \vee \overline{\mathcal{N}}^3_{e_{d_1}} \\
\overline{\mathcal{N}}^1_{e_{d_1}} &\triangleq (s=0) \wedge (s'=0) \wedge (\mathcal{N}_{e_{d_1}} \vee Unch(\langle \Theta_{e_{d_1}}, v_{e_{d_1}} \rangle) \\
\overline{\mathcal{N}}^2_{e_{d_1}} &\triangleq (s=0) \wedge (s'=1) \wedge \neg(\mathcal{N}_{e_{d_1}} \vee Unch(\langle \Theta_{e_{d_1}}, v_{e_{d_1}} \rangle) \\
\overline{\mathcal{N}}^3_{e_{d_1}} &\triangleq (s=1) \wedge Unch(\langle s, v_{d_1} \rangle) \\
\bar{v}_{e_{d_1}} &\triangleq \langle \Theta_{e_{d_1}}, v_{e_{d_1}}, v_{d_1}, s \rangle.
\end{aligned}
$$

The quantification and closure operators associated with $(E_{d_1})_{+v_{d_1}}$, M_{d_1}, and P_{d_1} are removed by applying Propositions 1 and 2 of [10]. Consequently, (10) can be written as

$$
\begin{aligned}
&\overline{Init}_{e_{d_1}} \wedge Init_{d_1} \wedge \Box[(\overline{\mathcal{N}}_{e_{d_1}} \vee Unch(\bar{v}_{e_{d_1}})) \\
&\quad \wedge (\mathcal{N}_{d_1} \vee Unch(\langle \Theta_{d_1}, v_{d_1} \rangle))]_{\langle \bar{v}_{e_{d_1}}, \Theta_{d_1}, v_{d_1} \rangle} \\
&\quad \Rightarrow Init_{p_{d_1}} \wedge \Box[\mathcal{N}_{p_{d_1}}]_{\langle v_{d_1} \rangle}.
\end{aligned}
$$

Proving the preceding formula requires demonstrating that each of the left-hand side sub-formulas separated by a disjunction implements the right-hand side formula. All but one of these sub-formulas can be easily shown to implement $\mathcal{N}_{p_{d_1}}$ using TLA axioms and proof rules. The final sub-formula which must implement $\mathcal{N}_{p_{d_1}}$ is $Init_{e_{d_1}} \wedge (s=0) \wedge \Box[\overline{\mathcal{N}}^1_{e_{d_1}} \wedge \mathcal{N}_{d_1}] \Rightarrow \mathcal{N}_{p_{d_1}}$. This sub-formula is similar to the one in (8), and its proof parallels the argument used in showing hypothesis (2b) for defender 1. Thus, we have shown that hypothesis (2a) holds for defender 1.

4) PROOF OF HYPOTHESES (2a) AND (2b) FOR DEFENDER 2

The proof of hypotheses (2a) and (2b) for defender 2 utilizes similar arguments to those used in the case of defender 1 and so is omitted.

Finally, hypothesis (3) is trivially satisfied by construction. Theorem 1 can then be invoked as all its hypotheses hold for the two-component system, leading to the desired conclusion. ∎

The following lemma constitutes the second step in verifying the global system property.

Lemma 4: Given the system shown in Fig. 1 with the specifications and environmental assumptions as given in Sections IV-D, IV-E, and IV-F, the following holds:

$$P_{d_1} \wedge P_{d_2} \Rightarrow P_{gl}.$$

Proof: First, given the specifications M_{d_1} and M_{d_2}, we can show that $\hat{P}_{v_2} = P_{v_2}$, $\hat{P}_{v_1} = P_{v_1}$, and $P_{e_1} = P_{e_2}$ by induction. Using these results and the specifications for P_{d_1} and P_{d_2}, the following holds:

$$
\begin{aligned}
P_{d_1} \wedge P_{d_2} &\Rightarrow \overline{Init}_{gl} \wedge \Box[\overline{\mathcal{N}}_{p_{gl}}]_{\langle v_{gl} \rangle}, \text{ where} \qquad (11) \\
\overline{Init}_{gl} &\triangleq t_1 = 1 \wedge t_2 = 1 \wedge \overline{\mathcal{N}}_{p_{gl}} \\
\overline{\mathcal{N}}_{p_{gl}} &\triangleq t_1 \le T_{max} \wedge t_2 \le T_{max} \wedge (\overline{\mathcal{N}}^1_{p_{gl}} \vee \overline{\mathcal{N}}^2_{p_{gl}}) \\
\overline{\mathcal{N}}^1_{p_{gl}} &\triangleq \|P_1 - P_{e_1}\| \le (\mathtt{D}_1 + \mathtt{D}_2 + \epsilon_1) \\
&\quad \wedge \|P_1 - P_{e_1}\| \ge (\mathtt{D}_1 - \mathtt{D}_2 - \epsilon_1) \\
&\quad \wedge \|P_2 - P_{e_2}\| \ge (\mathtt{D}_1 - \mathtt{D}_2 - \epsilon_1) \\
&\quad \wedge \|P_2 - P_1\| \ge (\mathtt{H}_{\mathtt{of}} - 2\mathtt{D}_2) \\
\overline{\mathcal{N}}^2_{p_{gl}} &\triangleq \|P_2 - P_{e_2}\| \le (\mathtt{D}_1 + \mathtt{D}_2 + \epsilon_1) \\
&\quad \wedge \|P_2 - P_{e_2}\| \ge (\mathtt{D}_1 - \mathtt{D}_2 - \epsilon_1) \\
&\quad \wedge \|P_1 - P_{e_1}\| \ge (\mathtt{D}_1 - \mathtt{D}_2 - \epsilon_1) \\
&\quad \wedge \|P_2 - P_1\| \ge (\mathtt{H}_{\mathtt{of}} - 2\mathtt{D}_2) \\
v_{gl} &\triangleq \langle P_{e_1}, P_{e_2}, P_1, P_2, t_1, t_2 \rangle.
\end{aligned}
$$

It is easy to verify that $\mathtt{D}_a$ and $\mathtt{D}_s$ satisfy the inequalities $\mathtt{D}_a \ge (\mathtt{D}_1 + \mathtt{D}_2 + \epsilon_1)$ and $\mathtt{D}_s \le (\mathtt{H}_{\mathtt{of}} - 2\mathtt{D}_2) \le (\mathtt{D}_1 - \mathtt{D}_2 - \epsilon_1)$. Combining (11) with these inequalities, we obtain

$$P_{d_1} \wedge P_{d_2} \Rightarrow P_{gl}.$$

∎

From Theorem 2 and Lemma 4, we conclude that $\models E_{gl} \wedge M_{d_1} \wedge M_{d_2} \Rightarrow P_{gl}$ holds, thereby demonstrating that the composed system satisfies the global system property.

V. CONCLUSION

This paper presents a case study that demonstrates how tools from compositional verification can be employed to design and perform reliability analysis of an unmanned multi-aircraft system. The case study involves two fixed-wing UAS that are jointly tasked to compromise an aerial encroacher. We adopt the compositional framework of [10] and show how the components of the multi-aircraft system can be designed such that the need for simulations in verifying the global system property is minimized. The main contribution of this work is to show how compositional reasoning can be effectively used to increase the reliability of complex multi-agent systems such as the unmanned multi-aircraft system considered herein.

APPENDIX

PROOF OF LEMMA 1

Throughout the proof, we use first order approximations to compute the positions of the virtual vehicle and the encroacher at time t_{k+1} given their positions and velocities at time t_k, i.e., $x_e(k+1) = x_e(k) + \tau v_{xe}(k)$ and so on. First, we will prove part 1 corresponding to $\text{Mode}_1 = 1$. At any time instant t_k, Algorithm 1 under $\text{Mode}_1 = 1$ computes the velocity of defender 1's virtual vehicle at t_k and then updates the position of the virtual vehicle. The virtual vehicle velocity components, denoted by v_{xv_1}, v_{yv_1}, and v_{zv_1}, are required to satisfy the following conditions:

$$
\begin{aligned}
&v^2_{xv_1} + v^2_{yv_1} + v^2_{zv_1} \le \mathtt{V}^2_{\mathtt{d}_{\max}} \qquad (12) \\
&(v_{xv_1} - v_{xe}(k) + dx(k)/\tau)^2 \\
&\qquad + (v_{yv_1} - v_{ye}(k) + dy(k)/\tau)^2 \\
&\qquad + (v_{zv_1} - v_{ze}(k) + dz(k)/\tau)^2 = (\mathtt{D}_1/\tau)^2. \qquad (13)
\end{aligned}
$$

Inequality (12) constrains the virtual vehicle velocity such that the resultant speed does not exceed $\mathtt{V}_{\mathtt{d}_{\max}}$, and equation (13) ensures that the separation distance between the

virtual vehicle and the encroacher at time t_{k+1} is $\mathtt{D}_1$. Inequality (12) defines a disk in the variables v_{xv_1} and v_{yv_1} with center (c_{1x}, c_{1y}) and radius r_1, where c_{1x}, c_{1y}, and r_1 are as defined in Algorithm 1. Similarly, equation (13) defines a circle with center (c_{2x}, c_{2y}) and radius r_2. These geometric interpretations will be helpful in this proof.

Since the proof is trivial for the case $|v_{ze}(k)| \leq \mathtt{V}_{\mathtt{zd}_{\max}}$, we directly proceed to the case where $\mathtt{V}_{\mathtt{zd}_{\max}} < |v_{ze}(k)| \leq \mathtt{V}_{\mathtt{ze}_{\max}}$. By construction, the virtual vehicle velocity components v_{xv_1} and v_{yv_1} computed in Algorithm 1 for $\text{Mode}_1 = 1$ lie on the circle defined by (13) and within the disk defined by (12). Therefore, to complete the proof, we need to show that for all values of $v_{xe}(k)$, $v_{ye}(k)$, and $v_{ze}(k)$ such that

$$v_{xe}(k)^2 + v_{ye}(k)^2 + v_{ze}(k)^2 \leq \mathtt{V}_{\mathtt{e}_{\max}}^2 \tag{14}$$

and $\mathtt{V}_{\mathtt{zd}_{\max}} < |v_{ze}(k)| \leq \mathtt{V}_{\mathtt{ze}_{\max}}$, the distance between the two centers (c_{1x}, c_{1y}) and (c_{2x}, c_{2y}), which is denoted by d_{12}, is always less than or equal to $r_1 + r_2$. Recognizing that $r_1 \leq r_2$, we need to show that $r_2 - r_1 \leq d_{12} \leq r_1 + r_2$ holds. The last condition states that the disk defined by (12) is never completely inside the circle defined by (13).

We now show that for all values of $v_{xe}(k)$, $v_{ye}(k)$, and $v_{ze}(k)$ satisfying (14) and $|v_{ze}(k)| \leq \mathtt{V}_{\mathtt{ze}_{\max}}$, $d_{12} \leq r_1 + r_2$ holds. Expanding this inequality results in

$$(v_{xe}(k) - dx(k)/\tau)^2 + (v_{ye}(k) - dy(k)/\tau)^2 \leq (r_1 + r_2)^2. \tag{15}$$

Define $r_3 = \sqrt{\mathtt{V}_{\mathtt{e}_{\max}}^2 - v_{ze}(k)^2}$. Geometrically, (14) and (15) define two disks in the variables $v_{xe}(k)$ and $v_{ye}(k)$; therefore, $d_{12} \leq r_1 + r_2$ if and only if the disk defined by (14) is completely inside the disk defined by (15), thereby resulting in the following condition: $(dx(k)/\tau)^2 + (dy(k)/\tau)^2 \leq (r_1 + r_2 - r_3)^2$, which can be further simplified to

$$(\mathtt{D}_1/\tau)^2 - (dz(k)/\tau)^2 - (r_1 + r_2 - r_3)^2 \leq 0. \tag{16}$$

For the given values of $\mathtt{D}_1$, $\mathtt{V}_{\mathtt{e}_{\max}}$, $\mathtt{V}_{\mathtt{d}_{\max}}$ $\mathtt{V}_{\mathtt{zd}_{\max}}$, $\mathtt{V}_{\mathtt{ze}_{\max}}$, and the constraint $|dz(k)| \leq \mathtt{H}_{\mathtt{of}}/2$, (16) can be easily shown to hold using Mathematica or any theorem prover that can handle real numbers. Herein, we use Mathematica to show (16).

Next, we show that $r_2 - r_1 \leq d_{12}$ holds. Expanding this inequality results in

$$(v_{xe}(k) - dx(k)/\tau)^2 + (v_{ye}(k) - dy(k)/\tau)^2 \geq (r_2 - r_1)^2. \tag{17}$$

$r_2 - r_1 \leq d_{12}$ holds if and only if the disk defined by (14) is completely outside the circle defined by the equation $(v_{xe}(k) - dx(k)/\tau)^2 + (v_{ye}(k) - dy(k)/\tau)^2 = (r_2 - r_1)^2$, which can be equivalently written as

$$(\mathtt{D}_1/\tau)^2 - (dz(k)/\tau)^2 - (r_2 - r_1 + r_3)^2 > 0. \tag{18}$$

Again, for the given values of the constants $\mathtt{D}_1$, $\mathtt{V}_{\mathtt{e}_{\max}}$, $\mathtt{V}_{\mathtt{d}_{\max}}$, $\mathtt{V}_{\mathtt{zd}_{\max}}$, $\mathtt{V}_{\mathtt{ze}_{\max}}$ and the constraint $|dz(k)| \leq \mathtt{H}_{\mathtt{of}}/2$, (18) can be easily shown using Mathematica. Thus, we have shown part (1a). From inspection of Algorithm 1, we see that part (1b) holds. To show (1c), we first observe that $\hat{dx}(k+1)^2 + \hat{dy}(k+1)^2 + \hat{dz}(k+1)^2 = \mathtt{D}_1{}^2 - \mathtt{H}_{\mathtt{of}}^2 + 2\hat{dz}(k+1)\mathtt{H}_{\mathtt{of}}$. Since $|dz(k)| \leq \mathtt{H}_{\mathtt{of}}/2$, we have $\mathtt{H}_{\mathtt{of}}/2 - \tau(\mathtt{V}_{\mathtt{ze}_{\max}} - \mathtt{V}_{\mathtt{zd}_{\max}}) \leq \hat{dz}(k+1) \leq 3\mathtt{H}_{\mathtt{of}}/2 + \tau(\mathtt{V}_{\mathtt{ze}_{\max}} - \mathtt{V}_{\mathtt{zd}_{\max}})$. Thus, we can write

$$\hat{dx}(k+1)^2 + \hat{dy}(k+1)^2 + \hat{dz}(k+1)^2 \geq \mathtt{D}_1{}^2 - 2\tau(\mathtt{V}_{\mathtt{ze}_{\max}} - \mathtt{V}_{\mathtt{zd}_{\max}})\mathtt{H}_{\mathtt{of}}.$$

For the values of $\mathtt{V}_{\mathtt{zd}_{\max}}$, $\mathtt{V}_{\mathtt{ze}_{\max}}$, and $\mathtt{H}_{\mathtt{of}}$, we have $\mathtt{D}_1{}^2 - 2\tau(\mathtt{V}_{\mathtt{ze}_{\max}} - \mathtt{V}_{\mathtt{zd}_{\max}})\mathtt{H}_{\mathtt{of}} > (\mathtt{D}_1 - \epsilon_1)^2$, or equivalently $\mathtt{D}_1{}^2 - 2\tau(\mathtt{V}_{\mathtt{ze}_{\max}} - \mathtt{V}_{\mathtt{zd}_{\max}})\mathtt{H}_{\mathtt{of}} > 19.3^2$, thus proving (1c).

The second part of the result for the case where $\text{Mode}_1 = 2$ can be analogously shown following similar steps as above and is therefore not provided here.

REFERENCES

[1] *Autonomy Community of Interest (COI) Test and Evaluation, Verification and Validation (TEVV) Working Group: Technology Investment Strategy 2015-2018*, Office Assistant Secretary Defense (Research and Engineering), Washington, DC, USA, 2015.
[2] RTCA, "DO-178C: Software considerations in airborne systems and equipment certification," RTCA, Washington, DC, USA, Tech. Rep., 2012.
[3] RTCA, "DO-254: Design assurance guidance for airborne electronic hardware," RTCA, Washington, DC, USA, Tech. Rep., 2000.
[4] D. A. Peled, *Software Release Methodology*. New York, NY, USA: Springer-Verlag, 2001.
[5] R. A. David and P. Nielsen, "Defense science board summer study on autonomy," Defense Sci. Board, Washington, DC, USA, Tech. Rep., 2016.
[6] B. Butka, S. Mandalapu, and C. Kilgore, "Advanced verification methods for safety-critical airborne electronic hardware," Federal Aviation Admin. William J. Hughes Tech. Center, Wisconsin, NJ, USA, Tech. Rep. DOT/FAA/TC-14/41, 2015.
[7] C. M. Jonker, V. Robu, and J. Treur, "An agent architecture for multi-attribute negotiation using incomplete preference information," *Auton. Agent Multi-Agent Syst.*, vol. 15, no. 2, pp. 221–252, Aug. 2007.
[8] G. Brat, E. Denney, D. Giannakopoulou, J. Frank, and A. Jonsson, "Verification of autonomous systems for space applications," in *Proc. IEEE Aerosp. Conf.*, Aug. 2006,
[9] D. Giannakopoulou, C. S. Păsăreanu, and H. Barringer, "Component verification with automatically generated assumptions," *Automated Softw. Eng.*, vol. 12, no. 3, pp. 297–320, Jul. 2005.
[10] M. Abadi and L. Lamport, "Conjoining specifications," *ACM Trans. Program. Lang. Syst. (TOPLAS)*, vol. 17, no. 3, pp. 507–535, May 1995.
[11] W. Dufrene, "Mobile military security with concentration on unmanned aerial vehicles," in *Proc. 24th Digit. Avionics Syst. Conf.*, Jan. 2006, pp. 8.D.3.1–8.D.3.8.
[12] S. Buerger, J. R. Salton, D. K. Novick, R. Fierro, A. Vinod, B. HomChaudhuri, and M. Oishi, "Reachable set computation and tracking with multiple pursuers for the ASAP counter-UAS capability," Sandia Nat. Lab, Albuquerque, NM, USA, Tech. Rep. SAND2016-9177C, 2016.
[13] S. Tolman and R. W. Beard, "Counter UAS using a formation controlled dragnet," in *Proc. Int. Conf. Unmanned Aircr. Syst. (ICUAS)*, Jun. 2017, pp. 1665–1672.
[14] R. Altawy and A. M. Youssef, "Security, privacy, and safety aspects of civilian drones: A survey," *ACM Trans. Cyber-Phys. Syst.*, vol. 1, no. 2, p. 7, 2017.
[15] Y. Son, H. Shin, D. Kim, Y.-S. Park, J. Noh, K. Choi, J. Choi, and Y. Kim, "Rocking drones with intentional sound noise on gyroscopic sensors," in *Proc. USENIX Secur. Symp.*, 2015, pp. 881–896.
[16] J. Selvaraj, G. Y. Dayanıklı, N. P. Gaunkar, D. Ware, R. M. Gerdes, and M. Mina, "Electromagnetic induction attacks against embedded systems," in *Proc. Asia Conf. Comput. Commun. Secur.-ASIACCS*, 2018, pp. 499–510.
[17] Center for Unmanned Aircraft Systems. (2019). *Small Aircraft Flight Encounters (SAFE) Data Repository*. [Online]. Available: https://sites.google.com/vt.edu/safe-repository/
[18] RTCA, "'DO-333: Formal methods supplement to DO-178C and DO-278A," RTCA, Washington, DC, USA, Tech. Rep., 2011.
[19] A. Pnueli, "In transition from global to modular temporal reasoning about programs," in *Logics and Models of Concurrent Systems*, 1985, pp. 123–144.

[20] R. Alur and T. A. Henzinger, "Reactive modules," *Formal Methods Syst. Des.*, vol. 15, no. 1, pp. 7–48, Jul. 1999.
[21] A. Megretski and A. Rantzer, "System analysis via integral quadratic constraints," *IEEE Trans. Autom. Control*, vol. 42, no. 6, pp. 819–830, Jun. 1997.
[22] L. Lamport, "The temporal logic of actions," *ACM Trans. Program. Lang. Syst. (TOPLAS)*, vol. 16, no. 3, pp. 872–923, May 1994.
[23] Hobby Express. (2017). *Senior Telemaster Plus*. [Online]. Available: http://www.hobbyexpress.com/senior_telemaster_plus_1034837_prd1.html
[24] J. R. Raol and J. Singh, *Flight Mechanics Modeling and Analysis*. Boca Raton, FL, USA: CNC Press, 2009.
[25] D. Muniraj, M. C. Palframan, K. T. Guthrie, and M. Farhood, "Path-following control of small fixed-wing unmanned aircraft systems with H_∞ type performance," *Control Eng. Pract.*, vol. 67, pp. 76–91, Oct. 2017.
[26] R. W. Beard and T. W. McLain, *Small Unmanned Aircraft: Theory and Practice*. Princeton, NJ, USA: Princeton Univ. Press, 2012.
[27] Federal Aviation Administration (FAA). *Code of Federal Regulations—Part 107*. [Online]. Available: https://www.ecfr.gov/cgi-bin/text-idx?SID=e331c2fe611df1717386d29eee38b000&mc=true&node=pt14.2.107&rgn=div5
[28] P. Gahinet and P. Apkarian, "A linear matrix inequality approach to H_∞ control," *Int. J. Robust Nonlinear Control*, vol. 4, no. 4, pp. 421–448, 1994.
[29] J. Lofberg, "YALMIP: A toolbox for modeling and optimization in MATLAB," in *Proc. IEEE Int. Conf. Robot. Autom.*, Mar. 2005, pp. 284–289.
[30] MOSEK ApS. (2018). *The MOSEK Optimization Toolbox for MATLAB Manual. Version 8.1.0.54*. [Online]. Available: https://docs.mosek.com/8.1/toolbox/index.html
[31] S. Gage, "Creating a unified graphical wind turbulence model from multiple specifications," in *Proc. AIAA Modeling Simulation Technol. Conf. Exhib.*, Aug. 2003.
[32] A. V. Oppenheim, A. S. Willsky, and S. Nawab, *Signals and Systems*. Upper Saddle River, NJ, USA: Prentice-Hall, 1996.
[33] M. C. Palframan, J. M. Fry, and M. Farhood, "Robustness analysis of flight controllers for fixed-wing unmanned aircraft systems using integral quadratic constraints," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 1, pp. 86–102, Jan. 2019.
[34] L. Lamport, "Hybrid systems in TLA$^+$," in *Hybrid Systems*, 1993, pp. 77–102.

DEVAPRAKASH MUNIRAJ received the B.E. degree in aeronautical engineering from the Madras Institute of Technology, Anna University, India, in 2010, and the Ph.D. degree in aerospace engineering from Virginia Tech, USA, in 2019. He worked as a Scientist with the Integrated Flight Control Systems Directorate, Aeronautical Development Agency, India, from 2010 to 2014. He is currently a Post-Doctoral Research Associate with the Department of Aerospace and Ocean Engineering, Virginia Tech. His current research interests include robust control design and formal verification of unmanned aircraft systems and autonomous underwater vehicles.

DANY ABOU JAOUDE received the B.E. degree in mechanical engineering from the American University of Beirut, in 2014, and the Ph.D. degree in aerospace engineering from Virginia Tech, in 2018. He is currently an Assistant Professor with the Mechanical Engineering Department, American University of Beirut, Lebanon. His areas of current research interest include distributed control, model reduction, and robustness analysis.

MAZEN FARHOOD received the M.S. and Ph.D. degrees in mechanical engineering from the University of Illinois at Urbana–Champaign, Urbana, IL, USA, in 2001 and 2005, respectively. Before joining Virginia Tech, Blacksburg, VA, USA, in 2008, he was a Scientific Researcher with the Delft Center for Systems and Control, Delft University of Technology, The Netherlands, and a Post-Doctoral Fellow with the School of Aerospace Engineering, Georgia Tech, Atlanta, GA, USA. He is currently an Associate Professor with the Kevin T. Crofton Department of Aerospace and Ocean Engineering, Virginia Tech. His current research interests include distributed control, motion planning and tracking along trajectories, model complexity reduction, and reliability analysis of unmanned aircraft system (UAS) flight control systems. He was a recipient of the National Science Foundation CAREER Award, in 2014.