JOURNAL OF AEROSPACE INFORMATION SYSTEMS
Vol. 13, No. 9, September 2016

Formal Verification for Mode Confusion in the Flight Deck Using Intent-Based Abstraction

Jayaprakash Suraj Nandiganahalli,* Sangjin Lee,† and Inseok Hwang‡
Purdue University, West Lafayette, Indiana 47907

DOI: 10.2514/1.I010393

As the flight deck has become highly automated, mode confusion between the pilot and the automation has emerged as an important issue for aviation safety. This paper presents a formal verification framework that can be used to efficiently detect a wide range of mode-confusion problems in the pilot–automation system and provide safety guarantees. To facilitate this, a novel formal modeling of the automation and pilot is proposed to efficiently verify the pilot–automation system. The automation of the aircraft is modeled as a deterministic hybrid system, and the pilot is modeled as an intent-based finite state machine. Due to the high dimension of the aircraft's continuous states and the large number of flight-mode combinations, formal verification of the hybrid system is computationally formidable, leading to the state-space explosion problem. To tackle this problem, a computationally efficient abstraction method for the hybrid model is proposed using intent inference, from which an intent-based finite state machine for the automation is obtained. The intent-based finite state machines for the automation and pilot are synchronously composed to systematically and comprehensively verify the pilot–automation system using the NuSMV model-checking tool. The proposed framework is demonstrated with two real pilot–automation interaction incidents/accidents.

I. Introduction

THE usage of autonomous systems in aircraft is in a period of continuous growth due to automation's potential to provide improved accuracy and efficiency. However, over-reliance on automation (e.g., autopilot, autothrottle) raises potential safety issues such as "mode-confusion" or "lack of mode awareness" incidents/accidents in the aircraft flight deck, which occur when the pilot loses his/her situational awareness about the current and future behavior of the aircraft. This could result in dysfunctional and unsafe pilot–automation interaction (e.g., unexpected altitude changes that go unnoticed by the pilot), as reported in the NASA Aviation Safety Reporting System reports and others [1–4]. Mode confusion happens due to a variety of reasons such as the complex logic of the automation, complexities in the pilot's behavior, and poorly represented information on the user interface (UI) [5]. This can lead to scenarios where the expectations of the pilot about the aircraft's actual behavior differ from the automation's intended behavior as observed via its UI.

Mode confusion has been studied in a number of ways such as simulation [6–8], data mining [9,10], human factors [2,5], formal methods such as model checking [11–14], etc. Although the simulation and testing of complex automation systems are more scalable than formal methods, they are rarely exhaustive, and thus could miss detecting potentially dangerous system failures in the pilot–automation system, including mode confusion, because these failures may occur under unexpected and infrequent combinations of conditions [15]. Data-driven approaches base their decisions of anomalies on the data itself and not on the models that generate the data, thus neglecting a lot of internal dynamical effects that can cause mode confusion. Sarter and Woods [5], in a series of studies about mode confusion, showed the limitations of the pilot's understanding of autoflight logic in dynamic situations and emphasized that "what is needed is a better understanding of how the machine operates, not just how to operate the machine." This led to a vast agreement that any further progress in systematic analysis and elimination of mode confusion must involve examining all the behaviors of a generalized description of the automation (e.g., a full fidelity formal model of the automation) and understanding the pilot's behavior with respect to his/her interaction with the automation (e.g., pilot's mental model) rather than examining just some of the behaviors of the real system (as with simulation or direct testing) [7,16] in order to guarantee trustworthy and safe aircraft operations. Formal methods have proven quite successful in situations where precise input–output models are defined by state variables and by transition logic that depends on environmental actions such as control inputs and disturbances. In this work, our focus is on detecting mode confusion by the formal methods approach, i.e., using a model checker that requires a formal mathematical model of the pilot–automation system in order to exhaustively search the system's operational state space for all initial states and possible inputs, as well as guarantee the existence or absence of a mode confusion. In our mode-confusion problem, the system comprises the automation, the pilot, and the interaction between them; thus, we need formal models of each of these components.

Realizing the aforementioned goal is challenging because of the following: First, the construction of proper automation and pilot models is not straightforward. Crow et al. [2] and Javaux [17] explored the use of finite state machines (FSMs) to describe the functional behaviors of the human operator and autopilot under both pilot-controlled and autonomous transitions for tactical and strategic decision-making. Rushby [12] studied mode confusion through formal verification by modeling the automation and pilot as FSMs. Degani [14] proposed a statechart formalism for mode-confusion detection by modeling the automation and the interface as FSMs, without explicitly modeling the pilot. Bass et al. [13] also explored a method for modeling and analyzing interactive hybrid systems, although at a very abstract level, using relational abstractions and demonstrating their method with an A-320 speed protection incident. They performed a bottom–up approach where very approximate models were initially constructed and used for mode-confusion analysis. If an interesting anomaly (i.e., a candidate mode confusion) was detected, then they checked whether it was an actual mode confusion or an artifact of poor modeling. If it was the latter, new constraints were added to the automation model. Thus, this approach required a lot of manual tuning to obtain sufficiently realistic automation models. Oishi et al. [18,19]

Presented as Paper 2016-0129 at the AIAA Infotech@Aerospace, San Diego, CA, 4–8 January 2016; received 13 April 2015; revision received 20 January 2016; accepted for publication 14 June 2016; published online 6 October 2016.

*Ph.D. Student, School of Aeronautics and Astronautics; jnandiga@purdue.edu. Student Member AIAA.
†Ph.D. Student, School of Aeronautics and Astronautics; lee997@purdue.edu. Student Member AIAA.
‡Associate Professor, School of Aeronautics and Astronautics; ihwang@purdue.edu. Associate Fellow AIAA.

explored the verification of the cockpit UI by developing a discrete abstraction of the hybrid model of the automation using reachability analysis and compositionally verifying the model with the finite state model of the UI. However, they did not explicitly model the pilot's behavior for verification, and hence could not detect mode-confusion scenarios explicitly. Sherry et al. [20] proposed a situation–goal–action input–output model of the automation and human operator, which is a goal-based model layered upon an extended FSM, to algorithmically find potential mode confusion.

Other human factor studies such as in [21,22] proposed task analytic models to capture the normative human operator behaviors during their interaction with the automation. Recently, Bolton and Bass [23] and Bolton [24] proposed an enhanced operator function model (EOFM) to capture observable manifestations of human behavior. The EOFM offers detailed discrete level modeling of how a human operator realizes his/her goal. The EOFM extends the operator function model to facilitate task models (i.e., those task analytic models concerned with capturing observable manifestations of human behavior, as opposed to cognitive models that are concerned with describing the cognitive process that drives the observable human behavior) to be evaluated with formal methods. This has been used in conjunction with finite state transition models of the automation and operator interface to verify pilot compliance/noncompliance during the before-landing checklist, diagnose failures of human–device interfaces, etc.

However, none of the aforementioned works focused on incorporating the continuous state dynamics of the aircraft into their formalism, and thus cannot account for the interaction between discrete and continuous dynamics in the decision-making process. This is important because there are instances of mode confusion that happen in the continuous states (e.g., undesired altitude change), as observed from previous studies by Nandiganahalli et al. [25] and Lee et al. [26]. This is crucial for detecting a range of mode-confusion scenarios such as parameter confusions, omission and commission errors, or sequences of mode confusions [27,28]. This is because mode confusion is not defined as "the divergence between the state of the mental model and the mode of the automation" but rather as "the divergence between the state of the mental model and the actual behavior of the aircraft as observed via its interface" [13]. Thus, mode confusion is about comparing the aircraft's actual behavior and the behavior expected by the pilot. In this sense, the range of problems encompassed by mode confusion involves both the aircraft's continuous and discrete behaviors rather than just the discrete mode-level behavior. Thus, capturing both the continuous and discrete aspects of the pilot–automation system is important when formally modeling the automation and pilot.

Second, most automated systems such as aircraft are cyberphysical systems, i.e., hybrid systems that use computational elements to control physical entities [19,29]. Verifying hybrid systems is computationally formidable because of the state-space explosion problem due to the large number of (infinite-dimensional) continuous state and flight-mode combinations [15]. Studies such as [18,30] solved the reachability analysis of hybrid systems using partial differential equations, which can seldom efficiently handle a hybrid system with more than five continuous variables [13]. Others have used approximation techniques on the hybrid model to enable tractable verification. Studies such as [31,32] approximated the continuous dynamics using ellipsoids or polytopes, whereas [33,34] used predicates to perform relational abstraction of the hybrid system. However, it is not clear how the aforementioned works can be formally adapted to allow heterogeneous information (as discussed previously) to be incorporated into the mode-confusion detection framework. Others [7,11] modeled only the automation without modeling the pilot, and they detected mode confusion through simulation of the flight deck reviewed by pilots and experts. Though the simulation approach is empirically useful, it cannot explore all the possible mode-confusion scenarios in a systematic way, and thus safety guarantees cannot be obtained. Thus, an efficient abstraction procedure is required to sufficiently map the hybrid system to a finite-dimensional system that is amenable to various kinds of automated analysis (e.g., model checking).

The main contributions of this paper are 1) the development of a formal model for the automation as a hybrid system that enables accurate detection of complex instances of confusions that occur sequentially and that happen not just in the flight modes but also in the continuous states, as well as a formal model for the pilot as an FSM with flight intents as its states to capture his/her decision-making; 2) a predicate-based intent abstraction technique for mapping an infinite-dimensional system to a finite-dimensional system using qualitative reasoning, facilitating efficient formal verification for mode-confusion detection using the NuSMV [35] model checker; 3) the direct comparison of the pilot's inferred intent/goal with the automation's inferred intent/goal to detect a potential mode confusion while summarizing the heterogeneous information; and 4) the potential use of the proposed framework for investigating systematic redesign solutions to make the pilot–automation interaction safer.

This paper is organized as follows: Sec. II discusses the proposed abstraction framework for the pilot–automation system. In Sec. III, the proposed algorithm is demonstrated with two real mode-confusion incidents/accidents. Section IV provides discussion and future work, and conclusions are presented in Sec. V.

II. Formal Modeling and Intent-Based Abstraction Framework for Mode-Confusion Detection

In this section, formal state-space models of the automation and pilot are discussed. The proposed approach involves developing modular, interacting formal models of the automation and pilot. The interacting dynamics of the automation's continuous and logical behaviors are described using a hybrid system. To enable feasible formal verification using model checking, the hybrid system is abstracted through intent inference to obtain an intent-based FSM while preserving the crucial information for mode confusion. On the other hand, the pilot's decision-making behavior is directly modeled as an intent-based FSM. These two intent-based FSMs are then synchronously composed and formally verified for a desired safety specification stated using action computation tree logic (ACTL) with the NuSMV model checker [35].

A. Formal Modeling of Automation: Hybrid System

A wide range of the actual aircraft behavior has a hybrid nature that cannot be accurately captured by other simpler dynamic models [13,18,29]. A hybrid model is a natural way to describe the interacting physical and logical dynamics of the aircraft automation, as well as describe the coupling between the discrete and continuous state-dependent guard conditions. This enables us to accurately detect confusions that occur sequentially because it is possible to precisely describe the continuous evolution of the aircraft between the occurrence of such mode confusions by using a hybrid model. Using a hybrid model, the aircraft's motion can be described by decomposing its behavior into a sequence of discrete flight modes along with the mode-specific continuous dynamics, as shown in Fig. 1 [25,29]. The flight modes refer to the aircraft automation's discrete states [e.g., vertical speed (V/S), flight level change (FLCH), and vertical navigation] that govern the continuous evolution of the aircraft motion (e.g., speed change and altitude change). A brief description of some flight modes of the automation's flight management system is given in Table 1. For example, the aircraft's motion of climbing and leveling off at a target altitude can be described by a flight-mode transition from the "V/S" mode to the "capture" mode, and to the altitude hold ("ALT HLD") mode. In each flight mode, the dynamics of the continuous state (in this case, the altitude rate $\dot{h}$) are described to be initially $\dot{h} > 0$ in the V/S mode, $\dot{h} > 0$ in the capture mode, and $\dot{h} = 0$ in the ALT HLD mode. Note from Fig. 1 and Table 1 that the V/S mode should be manually engaged to achieve a desired vertical speed, whereas the capture mode is automatically engaged to enable smooth leveling off.

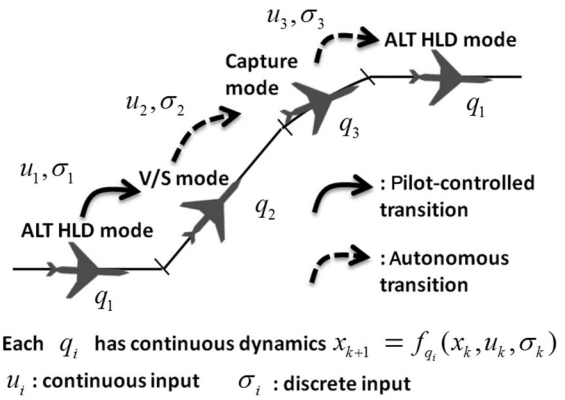

Fig. 1 Aircraft's motion shown as hybrid system in vertical dimension with altitude changes.

Formally, a discrete-time hybrid system is composed of the continuous state $x_k \in X \subseteq \mathcal{R}^n$ and discrete mode $q_k \in Q = \{1, 2, \ldots, N_q\}$, for which the evolutions (where k is the discrete time index) are described by the difference equation through a linear map $f(\cdot)$: $Q \times X \times U \to X$ and transition set-valued relation $R(\cdot)$: $Q \times X \times \Sigma \to 2^Q$, respectively, as

$$x_{k+1} = A_{q_k} x_k + B_{q_k} u_k \tag{1a}$$

$$q_{k+1} = R(q_k, x_k, \sigma_k) \text{if} \left[x_k^T \; \sigma_k^T\right]^T \in D^a(q_k, q_{k+1}) \tag{1b}$$

where Eq. (1) describes the interacting physical [Eq. (1a)] and logical [Eq. (1b)] dynamics of the aircraft automation's behavior; A_{q_k} is the system matrix; B_{q_k} is the continuous control input matrix; superscript T refers to the transpose of a vector; $\sigma \in \Sigma = \Sigma^a \cup \Sigma^p = (\{s_{\alpha\beta}\} \to \{\text{true}, \text{false}\})$ refers to the autonomous or forced discrete control inputs (superscripts a and p refer to the automation and pilot, respectively); $\{s_{\alpha\beta}\}$ denotes a discrete input signal set that enables the flight-mode transition from α to β; $u_k \in U = U^a \cup U^p$ denotes the continuous control inputs; and D^a: $Q \times Q \to 2^{X \times \Sigma}$ denotes the automation's guard set that represents the conditions under which the mode transitions happen as

$$D^a(\alpha, \beta) = \left\{ \left[x^T \sigma^T\right]^T \middle| x \in X, \sigma \in \Sigma, L_{\alpha\beta} x \le 0 \vee \sigma(s_{\alpha\beta}) = \text{true} \right\}, \quad \forall\, \alpha, \beta \in Q \tag{2}$$

In Eq. (2), $L_{\alpha\beta} x \le 0$ denotes a linear guard structure where some linear combination of continuous states x satisfying certain preset conditions trigger a flight-mode transition from flight mode α to flight mode β. Here, $L_{\alpha\beta}$ are trinary variables that take the value -1, 0, or 1, selecting certain states x over others. For example, if $v > v_{\text{maxspeed}}$ holds true (where v is the aircraft's airspeed, and v_{maxspeed} denotes maximum allowed airspeed in the configuration), which can be rewritten in matrix form with $L_{\alpha\beta} = -1$, $x = (v - v_{\text{maxspeed}})$ as $L_{\alpha\beta} x < 0$, in flight mode $\alpha = \text{V/S}$, then an automatic transition is enabled to a flight mode $\beta =$ open climb. Additionally, the discrete transition from flight mode $\alpha =$ ALT HLD to flight mode $\beta = \text{V/S}$ is enabled when the pilot applies a discrete input $s_{\alpha\beta}$ by pressing the V/S button and setting the desired vertical speed on the mode control panel (MCP).

Due to the high dimension of the aircraft's continuous states and a large number of flight-mode combinations, performing model checking on the aforementioned hybrid model of the automation is infeasible in general. To tackle this problem, the hybrid system is abstracted to a finite-dimensional FSM with flight intents as its states using intent inference such that the critical information about the mode confusion is retained. A flight intent is defined as an abstract state that captures the causal relationship between the aircraft's observed behavior and the automation's or the pilot's control commands, and it describes the immediate goal of the automation and pilot. Thus, assigning situation–action pairs with goal labels (i.e., flight intents) enables us to manage the complexity of the pilot–automation system for mode-confusion purposes [20]. Since intents are not directly available, we need to initially construct an intent set; then, we need to infer them using information such as the aircraft's continuous states, flight modes, and automation's and pilot's inputs.

1. *Construction of Flight Intent Set*

The motivations to introduce an abstract discrete intent state (in place of using the automation's continuous states and discrete flight modes) are threefold. First, the mode confusion is characterized by the divergence between the pilot's immediate goal/intent for the actual behavior of the automation and the goal of the automation, and thus happens in the intention domain. Second, using flight intents as the discrete states of the automation and pilot model, the automation's and pilot's complex behaviors can be described at a higher level (than the continuous or discrete aircraft states) and directly compared [20]. Third, an intent-based abstraction of the hybrid model enables us to put forward a coherent, computationally efficient and descriptive formal framework for mode-confusion detection, where the intents belong to a smaller cardinal set, unlike the infinite-dimensional continuous states.

Table 1 Generic flight modes (i.e., automation's discrete states)

Flight modes	Description
V/S	Pitch regulation by elevator to achieve desired vertical speed
Capture	Pitch regulation by elevator to smoothly level off to target altitude
ALT HLD	Maintain target altitude
HDG (heading)	Maintain magnetic heading
FLCH	Maintain airspeed by pitching up or down
SPD	Adjust thrust to maintain airspeed
THR idle	Engine thrust decreases to idle power

Table 2 Construction of intent set (last row refers to flight intents)[a]

Autothrottle flight modes	Autopilot flight modes	
Speed S	Lateral L	Vertical V
SPD	HDG SEL	V/S
THR (CLB or idle)	TRK	ALT HLD
A FLOOR	LOC	FLCH
...	...	...
{Accelerate, decelerate, constant speed}	{Turn left, turn right, constant heading}	{Climb, descend, constant altitude}

[a]Note that A FLOOR denotes Alpha floor, HDG SEL denotes heading select, TRK denotes track, and LOC denotes localizer.

In this sense, flight intents are used to address the safety issue effectively at a higher level than the continuous states and flight modes by inferring the cause that makes the automation or pilot issue necessary control commands. It is important to note that, in the realm of the cockpit under a given situation, only finite intents are relevant, as the intents are usually constrained by flight manuals, standard procedures, and Air Traffic Control (ATC) regulations [36]. The flight intent set is constructed by noting that the aircraft's behavior is induced by the flight modes of the automation (i.e., autothrottle and autopilot modes). The modes of the autothrottle govern the speed behavior of the aircraft, whereas the autopilot modes govern the lateral and vertical behaviors of the aircraft through inputs from the mode control panel (as shown in Table 2). More formally, the flight intent set $\{^iI\} \in \mathcal{I}$ given in Eq. (3) is constructed by performing a sign-based predicate abstraction of the continuous state rates along each of the speed S, lateral L, and vertical V dimensions in each flight mode using a set of three Boolean predicates and qualitative reasoning [37,38]. The aircraft's continuous state derivative is mapped into either less than zero, equal to zero, or greater than zero regions, i.e., ${}_l\mathbb{I} = {}_l\mathbb{I}_1 \cup {}_l\mathbb{I}_2 \cup {}_l\mathbb{I}_3$, $l = S, L, V$, where

$$
\begin{aligned}
{}_l\mathbb{I}_1 &= \{(\dot{x}_l, q_l) | \dot{x}_l < -\epsilon_r, \qquad q_l \in R(q, x, \sigma)\} \\
{}_l\mathbb{I}_2 &= \{(\dot{x}_l, q_l) | -\epsilon_r \le \dot{x}_l \le \epsilon_r, \qquad q_l \in R(q, x, \sigma)\} \\
{}_l\mathbb{I}_3 &= \{(\dot{x}_l, q_l) | \dot{x}_l > \epsilon_r, \qquad q_l \in R(q, x, \sigma)\}, \quad \forall\, q \in Q, \qquad x \in X, \qquad \sigma \in \Sigma
\end{aligned}
$$

$\cup$ is the union operator; and ϵ_r is a small positive number to account for uncertainties, by a technique known as parallel composition [39]. The flight intent set $\mathcal{I}$ is then compositionally constructed to generate a total of 27 three-tuple intent elements as

$$
\begin{aligned}
\mathcal{I} &= \{^1I^2, I, \ldots, {}^{27}I\} \\
{}^iI &= ({}^i_S\mathbb{I}, {}^i_L\mathbb{I}, {}^i_V\mathbb{I}), \qquad i \in \{1, 2, \ldots, 27\} \\
{}^i_S\mathbb{I} &\in \{\text{Accelerate, Decelerate, Constant Speed}\} \\
{}^i_L\mathbb{I} &\in \{\text{Turn Left, Turn Right, Constant Heading}\} \\
{}^i_V\mathbb{I} &\in \{\text{Climb, Descend, Constant Altitude}\}
\end{aligned} \tag{3}
$$

By the preceding construction, the tactical behavior of the aircraft can be overapproximated to a discrete mutually exclusive intent set (with cardinality of the set, at most, 27) sufficient for mode-confusion detection. For example, the speed v of an aircraft is abstracted into $\dot{v} > 0$ (accelerate intent), $\dot{v} < 0$ (decelerate intent), and $\dot{v} = 0$ (constant speed intent). Figure 2 describes the same vertical motion of the aircraft as in Fig. 1 but in the intent domain where flight intent transitions from the constant altitude intent to the climb intent, followed by transitioning back to the constant altitude intent.

The proposed abstraction procedure for the hybrid model of the automation is explained in the following, as in Fig. 3.

2. *Abstraction of Domain*

To succinctly describe the aircraft's motion for the purpose of mode-confusion detection, the hybrid states $X \times Q$ are mapped to the intent domain $\mathcal{I}$ by a function z: $X \times Q \to \mathcal{I}$, which infers the intent $I^a_k \in \mathcal{I}$ based on the sign of the continuous state derivatives (i.e., $\dot{x} < -\epsilon_r, -\epsilon_r \le \dot{x} \le \epsilon_r, \dot{x} > \epsilon_r$, where ϵ_r is a small positive number to account for uncertainties) for each discrete flight mode q_k. This is mathematically expressed as

$$
I^a_k = z(x_k, q_k), \qquad I \in \mathcal{I} \tag{4}
$$

For example, if $(\dot{h}_k > 0) \wedge (q_k \coloneqq \text{V/S mode})$, then $I^a_k = \text{climb}$, where $\wedge$ is the logical AND operator. Thus, the abstracted model for the hybrid system M is described in the intent domain $\mathcal{I}$, which is much smaller compared to the original infinite-dimensional domain. Thus, the state-space explosion problem is effectively addressed.

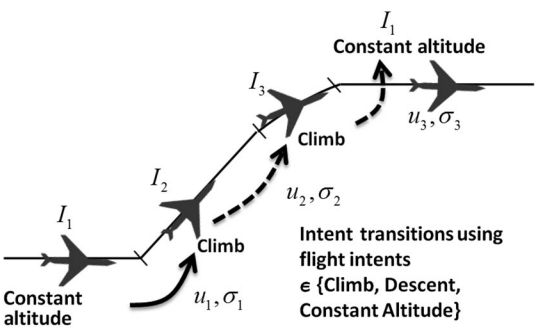

Fig. 2 Aircraft's motion shown as flight intent transitions in the vertical dimension.

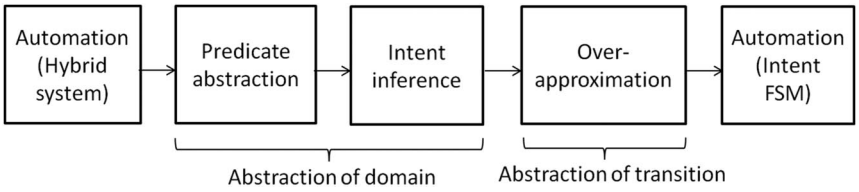

Fig. 3 Schematic of the proposed abstraction framework for the hybrid model.

3. Abstraction of Transition

It should, however, be noted that, in addition to the preceding domain abstraction, the transition relations must also be abstracted to obtain an intent-based FSM. Such an abstracted intent-based finite state model $\bar{M}^a$ of the automation describes the evolution of flight intents of the automation (for which the transition map depends on the aircraft's continuous states) and the control inputs that satisfy the guard condition [as in Eq. (2)], and it can be verified using a discrete model checker such as the NuSMV. The abstracted transition relation $\bar{R}$: $\mathcal{I} \times \Sigma \to \mathcal{I}$ is then given using Eq. (4) as

$$\begin{aligned} I^a_{k+1} &= \bar{R}(I^a_k, \sigma_k) \\ &= \bar{R}(z(x_k, q_k), \sigma_k) \\ &= \begin{cases} z((A_\beta x_k + B_\beta u_k), \beta) & \text{if } [x_k^T \ \sigma_k^T]^T \in D^a(\alpha, \beta), \quad \forall\, \beta \in Q \\ I^a_k & \text{else} \end{cases} \end{aligned} \tag{5}$$

where the continuous state propagation is performed according to Eq. (1a). The form of the abstracted transition relation $\bar{R}$ indicates that the next inferred intent state depends on the current (intent) state and inputs (i. e., control actions and hybrid states satisfying guards). Put another way, depending on the aircraft's current state (continuous and discrete) and pilot's control inputs, the intent at time $k+1$ could be either the same as the intent at time k or it could transition to a new intent according to the intent map given in Eq. (4) and hybrid dynamics and linear guard given in Eqs. (1) and (2), respectively. Hence, unlike most abstraction methods that often introduce nondeterminism in the abstracted models, this is not the case in our paper for mode-confusion detection. This is because the proposed intent-based abstraction is basically a predicate-based "renaming" of hybrid states into a three-valued abstract "intent" domain {negative, zero, positive} as given by the z map defined using qualitative reasoning of the first-order continuous state rate information. Thus, the proposed abstraction avoids introducing explicit nondeterminism in the intent states in the three dimensions (i.e., speed, lateral, and vertical).

Thus, the formal abstracted model of the automation is given as a deterministic intent-based FSM: $\bar{M}^a = (\bar{\Psi}^a, \bar{\Psi}^a_0, \bar{\Xi}^a, \bar{D}^a, \bar{R}^a)$, where $\bar{\Psi}^a \subseteq \mathcal{I}$ is a set of discrete intent states; $\bar{\Psi}^a_0$ is a set of initial intent states over $\bar{\Psi}^a$; $\bar{\Xi}^a \subseteq X \times Q \times \Sigma \times U$ is a set of hybrid actions; $\bar{D}^a$: $\bar{\Psi}^a \times \bar{\Psi}^a \to 2^{\bar{\Xi}^a} \cup \{\text{TRUE}\}$ is a guard set; and $\bar{R}^a$: $\bar{\Psi}^a \times \bar{\Xi}^a \to \bar{\Psi}^a$ is a set of discrete intent transition relations that map the automation's intent at time k to that at $k+1$, as in Eq. (5).

B. Formal Modeling of Pilot: Intent-Based FSM

Recall that flight-mode situational awareness problems such as mode confusion happen in the perception/intention domain of the pilot. Also, the pilot's behavior (and thus the corresponding pilot's intent) is a time-and-action-constrained complex task [6], which is usually constrained by the flight plans, ATC advisories, flight procedures, and aircraft dynamics. Furthermore, the formal model of the pilot's expectations/intentions must conform to the constraints imposed by the UI, which encodes various flight behaviors in terms of simple knob settings to facilitate vertical, lateral, and speed motions of the aircraft. Thus, the pilot model that we are looking for should have rich formal syntax and semantics that enable formal verification and, at the same time, be neither too complicated nor too simplistic so that it cannot capture important characteristics of the mode-confusion problem.

Since the pilot's expected aircraft behaviors usually follow a goal-directed policy transition triggered by his/her control commands [26,40], and mode confusion is a safety issue that happens as a conflict between the perception/goal of the pilot about the aircraft's expected behavior and the aircraft's actual behavior (and occurs at a level higher than the underlying flight modes and continuous states of an aircraft), in this paper, an intent-based finite state machine model of the pilot behavior is presented as effective and suitable for a range of mode-confusion problems. In this sense, this paper describes the pilot's behavior as the discrete transitions between his/her immediate goals/intent states $I^p \in \mathcal{I}$: triggered due to both input and to state-dependent guard conditions along the speed, lateral, and vertical dimensions. For example, the pilot-intent FSM is constructed based on the desired task the pilot tries to achieve when he/she decides the desired flight behavior (i.e., flight intent, e.g., $I^p_{k+1} = \text{climb}$) using the recent flight behavior (e.g., $I^p_k = \text{constant altitude}$), the aircraft's continuous states (e.g., $x_k = -\epsilon_r \le \dot{h}_k \le \epsilon_r$, where ϵ_r is a small positive number), and flight modes (e.g., $q_k \coloneqq$ ALT HLD mode). Accordingly, the pilot issues commands to the automation through the MCP (e.g., $\sigma^p_k \coloneqq$ the pilot either noses up using the control yoke or sets a higher target altitude H_s on the MCP) to achieve his/her desired mode transition (e.g., performs a climb maneuver due to satisfaction of the guard $H_s > h_k$).

If we consider the pilot's control behavior, it is often found that he/she appropriately switches between certain simple primitive goals/intents using the speed, lateral, and vertical dimensional buttons on the MCP to achieve his/her expected behavior of the aircraft [41]. For example, consider an aircraft maintaining a constant altitude when the pilot turns up the altitude knob on the MCP and engages the V/S knob, expecting the aircraft to perform a climb maneuver. Under this condition, it can be inferred that the pilot-intent state transitions to a different intent state (e.g., transitions from "constant altitude" intent to "climb" intent). The switching/transition between primitive flight intents, given as in Eq. (3), can approximate the pilot's decision-making behavior about the aircraft's expected tactical behavior while incorporating both the continuous and discrete information for mode-confusion purposes. Hence, the pilot is formally modeled as an intent-based FSM to describe certain aspects of human cognition, such as his/her summary predictions of what will happen, which is useful in detecting the mode confusion [13]. Such an intent-based model for the pilot has also been successfully used to study several other mode-confusion cases in related works [25–27,42], although not for formal verification purposes as in this paper. To mathematize this notion, the pilot-intent transitions are given by a formal linear guard for the pilot's decision making. The guard describes the aircraft state conditions presumed by the pilot, the pilot's inputs, and the pilot's understanding of the automation logic that, together, enable the mode transition. The mathematical description of the automation's guard structure as perceived by the pilot is as follows:

$$\bar{D}^p(i,j) = \left\{ \left[x^T \sigma^T u^T \right]^T \middle| x \in X, \sigma \in \Sigma, u \in U, K_{ij} x \le 0 \vee \sigma(w_{ij}) = \text{true} \right\}, \quad \forall\, \alpha, \beta \in Q \tag{6}$$

where superscript T refers to the transpose of a vector; $\{w_{ij}\}$ denotes a discrete input signal set that enables the flight intent transition from ${}^iI$ to ${}^jI$, as perceived by the pilot; and $K_{ij}x$ captures how well the pilot knows the dependence of the automation's transition logic on the continuous states to enable his/her intent transition from ${}^iI$ to ${}^jI$, as given in Eq. (3). Here, K_{ij} are trinary variables that take the value -1, 0, or 1, selecting certain states x over others; thus, $K_{ij}x \le 0$ denotes a linear guard condition for a pilot-intent state transition. For example, consider an aircraft maintaining a constant altitude when the pilot turns up the altitude knob on the MCP to increase the h_{set} value. Then, if $h < h_{\text{set}}$, where h is the aircraft's altitude, holds true [which can be rewritten in matrix form with $K_{ij} = 1, x = (h - h_{\text{set}})$, as $K_{ij}x \le 0$], the pilot-intent state transitions to a different intent state (e.g., transition from constant altitude intent to climb intent). These parameters of the pilot model could be determined from flight simulations/expert opinions. Then, Eq. (7) describes the pilot's intent transition logic where his/her intent (or an immediate goal) transitions from ${}^iI$ at time k to ${}^jI$ at time $k+1$. The flight intent can change when either the aircraft's continuous state approaches a certain set target/threshold value or due to the pilot's control inputs [as stated in Eq. (6)]. Additionally, the flight intent can remain the same (i.e., a default/self-transition from ${}^iI$ at time k to ${}^iI$ at time $k+1$) if no conditions in Eq. (6) hold true. Note that, generally, the pilot can exercise a nondeterministic choice due to both the effect of feedback, i.e., the transitions in pilot behaviors depend on environmental conditions such as aircraft states and ATC clearances, which are external to the pilot, and his/her control actions, e.g., either turn up or turn down the ALT knob, and dial a nondeterministic value into ALT window:

$$I^p_{k+1} = \begin{cases} {}^jI & \text{if } \left[x_k^T \sigma_k^T \right]^T \in \bar{D}^p(i,j), \quad \forall\, j \in \{1, 2, \ldots, 27\} \\ {}^iI & \text{else} \end{cases} \tag{7}$$

Thus, the formal deterministic intent-based FSM model of the pilot $\bar{M}^p$ is similarly given by replacing the superscript "a" in $\bar{M}^a$ with "p". It is worth pointing out that a typical cause for mode confusion is (as observed from the aforementioned automation and pilot models) $\bar{D}^a \neq \bar{D}^p$.

C. Safety Specification

The proposed framework allows mode confusion to be detected by directly comparing the pilot's inferred intent with that of the automation. This is formally given as a safety specification $\bar{\phi} :=$ AG nodivergence, where nodivergence is a binary decision variable for an absence of intent mismatch between the automation and pilot, expressed in temporal logic formalism using action computation tree logic with standard spatial and temporal boolean operators [43] as follows:

$$\text{AG nodivergence, where nodivergence} := \left(\text{automation.intentstate } I^a_k = \text{pilot.intentstate } I^p_k \right) \tag{8}$$

Note that $\bar{\phi} :=$ AG nodivergence literally means always globally nodivergence. The spatial and temporal operator AG instructs the model checker to explore all possible reachable states of the formal model into the future under the given initial condition and input set to determine if there exists no mismatch between the automation's and pilot's intent states in the abstracted pilot–automation system. In this sense, if $\bar{\phi}$ is found to be false any time along the exploration path, then we will have found an instance of mode confusion.

D. Verification Process

Finally, since the pilot's and automation's decision making are sequential in nature and depend on each other's input/output relationship, the formal models of the pilot and automation are synchronously combined to capture the interaction between the automation and pilot. Thus, the abstracted pilot–automation system (a Kripke model $\bar{M}^{ap}$) is obtained by performing (an event-triggered) synchronous–reactive composition [13] of the automation-intent FSM $\bar{M}^a$ and the pilot-intent FSM $\bar{M}^p$ for the desired specification $\bar{\phi} :=$ AG nodivergence, as summarized in Fig. 4. If all the reachable states upon exhaustive state-space exploration under all possible inputs of $\bar{M}^{ap}$ satisfy all the verification formulas of $\bar{\Phi} := \{\bar{\phi}\}$ (i.e., $\bar{M}^{ap} \vDash \bar{\Phi}$), then it is guaranteed that the pilot–automation interaction system $\bar{M}^{ap}$ is correct for the desired safety specification $\bar{\Phi}$. If there is a violation of any safety specification, then the model checker (e.g., the NuSMV) will provide a counterexample with a trace as to what led to that counterexample. The entire formal verification process is summarized in Fig. 4.

III. Demonstration of the Proposed Algorithm

In this section, the proposed method is demonstrated with two real mode-confusion examples.

A. Example 1: Kill-the-Capture Incident

1. Description of Incident

The "kill-the-capture" incident of 1989 is depicted in Fig. 5. In this incident [1], the Boeing-737 aircraft was initially climbing to reach a set altitude of $H_f = 27{,}000$ ft when it autonomously transitioned into the capture mode at $H_c = 25{,}000$ ft. Then, having received a new ATC instruction at 26,500 ft to descend to $H_s = 24{,}000$ ft, the pilot turned down the MCP knob to a new target altitude of 24,000 ft. Based on the

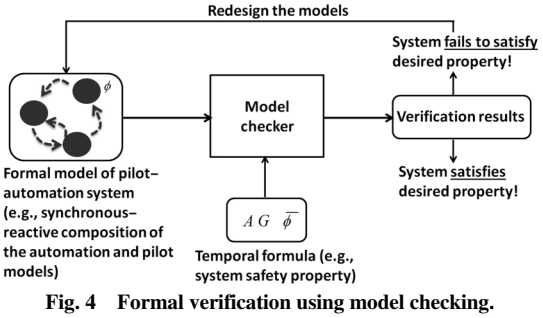

Fig. 4 Formal verification using model checking.

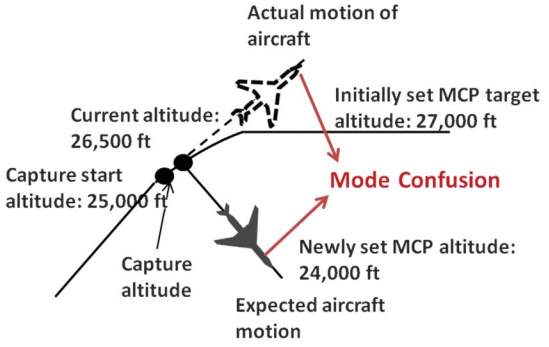

Fig. 5 Kill-the-capture mode-confusion incident (black continuous and broken lines indicate altitude).

autopilot logic, the aircraft autonomously transitioned to the vertical speed free climb mode, which was parameterized only by the vertical speed setting and unrestricted by any target altitude constraint. This resulted in the aircraft climbing unconstrained instead of the pilot-expected descent to 24,000 ft, causing the mode confusion. Unfortunately, the pilot was unaware of this autonomous mode transition from his/her perceived automation logic and the information presented on the UI. It should be noted that the automation in this incident was working correctly according to its designed logic, but the incorrect interaction with the pilot caused the mode-confusion problem.

2. Application of the Proposed Algorithm

The methodology for mode-confusion detection follows the proposed framework in the previous section.

a. Formal Model of the Automation: The hybrid system model for the automation that describes the aircraft's behavior for the kill-the-capture incident is discussed here. The different flight modes $q \in Q = \{1, 2, 3\}$ and their respective continuous dynamics $x_{k+1} = A_q x_k$, where $x_k = [h_k \dot{h}_k]^T$ (h and $\dot{h}$ are the aircraft's altitude and altitude rate, respectively) and A_q are 2×2 system matrices, are given in the following [26,42]:

1) V/S mode, $q = 1$:

$$\begin{bmatrix} h_{k+1} \\ \dot{h}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T_s \\ 0 & 1 \end{bmatrix} \begin{bmatrix} h_k \\ \dot{h}_k \end{bmatrix} \tag{9}$$

2) Capture mode, $q = 2$:

$$\begin{bmatrix} h_{k+1} \\ \dot{h}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T_s \\ \gamma^2 T_s & 1 \end{bmatrix} \begin{bmatrix} h_k \\ \dot{h}_k \end{bmatrix} + \begin{bmatrix} 0 \\ -\gamma^2 T_s H_f \end{bmatrix} \tag{10}$$

3) ALT HLD mode, $q = 3$:

$$\begin{bmatrix} h_{k+1} \\ \dot{h}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} h_k \\ \dot{h}_k \end{bmatrix} \tag{11}$$

where H_f, T_s, and γ denote the MCP target altitude set by the pilot, the sampling rate, and the capture rate, respectively.

The discrete mode transitions, which could be either autonomous or pilot triggered upon the satisfaction of the guard condition $D^a(\alpha, \beta)$, $\forall\ \alpha, \beta \in Q = \{1, 2, 3\}$ for the kill-the-capture incident, are given in the following. Note that the autopilot can do more complex cases, but to demonstrate the proposed method, the example considers a simple yet illustrative case. For example, an autonomous flight-mode transition from the V/S mode to the capture mode is triggered when the aircraft reaches the capture altitude, i.e., $h_k = H_c$:

$$\begin{aligned}
D^a(1,1) &= X \times \Sigma \setminus (D^a(1,2) \cup D^a(1,3)) \\
D^a(1,2) &= \left\{ \left[[h\ \dot{h}] \sigma^p\ \sigma^a \right]^T \middle| \operatorname{sgn}(\dot{h})(h - H_c) \geq 0 \right\} \\
D^a(1,3) &= \varnothing \\
D^a(2,1) &= \left\{ \left[[h\ \dot{h}] \sigma^p\ \sigma^a \right]^T \middle| H_s < H_c < h \vee \sigma^p = \sigma_3^p \right\} \\
D^a(2,2) &= X \times \Sigma \setminus (D^a(2,1) \cup D^a(2,3)) \\
D^a(2,3) &= \left\{ \left[[h\ \dot{h}] \sigma^p\ \sigma^a \right]^T \middle| h - H_f = 0 \right\} \\
D^a(3,1) &= \left\{ \left[[h\ \dot{h}] \sigma^p\ \sigma^a \right]^T \middle| h \neq H_f \vee \sigma^p = \sigma_3^p \right\} \\
D^a(3,2) &= \varnothing \\
D^a(3,3) &= X \times \Sigma \setminus (D^a(3,1) \cup D^a(3,2))
\end{aligned} \tag{12}$$

where superscript T refers to the transpose of a vector; $\setminus$ denotes the set difference operator; $\cup$ is the set union operator; sgn is the sign function; and $\varnothing$ denotes the empty set. In Eq. (12), $\sigma^p = \{\sigma_1^p, \sigma_2^p, \sigma_3^p\} \in \Sigma^p$ refers to the pilot's discrete inputs, where σ_1^p denotes the pilot turning the ALT knob to initial target altitude H_f, σ_2^p refers to the pilot turning the ALT knob to newly set altitude H_s, and σ_3^p refers to the V/S button engagement by the

pilot: $\sigma^a \in \Sigma^a$ is the automation's discrete input that refers to the capture altitude H_c computed by $H_c = H_f - \text{sgn}(\dot{h})\zeta$, where ζ is a design threshold.

b. Appropriate Flight Intent Set: The three-dimensional vertical intent state space (only the aircraft's vertical motion is relevant for this incident) for the kill-the-capture incident is given as follows:

$$I \in \mathcal{I} = \{\text{Climb, Descend, Constant Altitude}\} \tag{13}$$

Then, the intent-based FSM model of the automation can be obtained according to the intent transition logic in Eq. (5) for the preceding hybrid model, starting from the V/S mode and the climb intent, and is shown in Fig. 6. The following parameters are considered for initiating and running the algorithm [26,39]: $H_f = 27{,}000$ ft, $H_s = 24{,}000$ ft, $H_c = 25{,}000$ ft, initial altitude $h_0 = 24{,}000$ ft, initial altitude rate $\dot{h}_0 = 50$ ft/s, $\epsilon_r = 1$ ft/s, and $\zeta = 1000$ ft. The capture rate constant is taken as $\gamma = 0.2$, and the sampling time is taken as $T_s = 5$ s, assuming surveillance radar/mode C transponder is used for measurement (if automatic dependent surveillance-broadcast is assumed, update interval can be taken as $T_s = 1$ s, and this restriction can be relaxed during mechanization of the algorithm).

c. Formal Model of the Pilot: The pilot's intent-based FSM is constructed with the pilot's perceived guard conditions $\bar{D}^p(i,j), \forall\ i,j = \{1,2,3\}$ (where 1 denotes climb, 2 denotes descend, and 3 denotes constant altitude), as given in the following and shown in Fig. 7 (only important transitions are indicated), using the aircraft's continuous states and flight modes, along with the pilot's inputs:

$$\begin{aligned}
\bar{D}^p(1,1) &= X \times \Sigma \setminus (\bar{D}^p(1,2) \cup \bar{D}^p(1,3)) \\
\bar{D}^p(1,2) &= \left\{ \left[[h\ \dot{h}] \sigma^p\ \sigma^a \right]^T \middle| h > H_s \wedge \sigma^p = \sigma^p_3 \right\} \\
\bar{D}^p(1,3) &= \left\{ \left[[h\ \dot{h}] \sigma^p\ \sigma^a \right]^T \middle| |h - H_f| \le \delta \right\} \\
\bar{D}^p(2,1) &= \left\{ \left[[h\ \dot{h}] \sigma^p\ \sigma^a \right]^T \middle| h < H_s \wedge \sigma^p = \sigma^p_3 \right\} \\
\bar{D}^p(2,2) &= X \times \Sigma \setminus (\bar{D}^p(2,1) \cup \bar{D}^p(2,3)) \\
\bar{D}^p(2,3) &= \left\{ \left[[h\ \dot{h}] \sigma^p\ \sigma^a \right]^T \middle| |h - H_f| \le \delta \right\} \\
\bar{D}^p(3,1) &= \left\{ \left[[h\ \dot{h}] \sigma^p\ \sigma^a \right]^T \middle| h < H_s \wedge \sigma^p = \sigma^p_3 \right\} \\
\bar{D}^p(3,2) &= \left\{ \left[[h\ \dot{h}] \sigma^p\ \sigma^a \right]^T \middle| h > H_s \wedge \sigma^p = \sigma^p_3 \right\} \\
\bar{D}^p(3,3) &= X \times \Sigma \setminus (\bar{D}^p(3,1) \cup \bar{D}^p(3,2))
\end{aligned} \tag{14}$$

where δ is a constant parameter denoting the distance within which the altitude capture happens as perceived by the pilot. The pilot's intent transitions are obtained according to Eq. (7). For example, the pilot's model starts in the climb intent state, i.e., $I^p_k = \text{climb}$. If the guard $(H_s < h_k)$

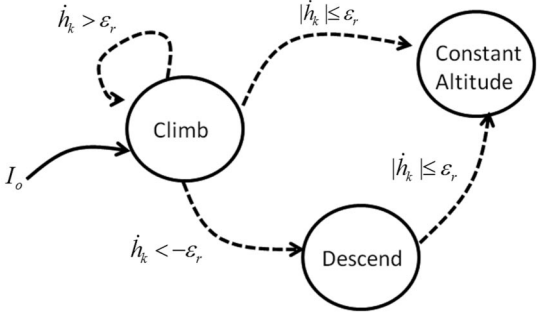

Fig. 6 Automation's intent FSM for the kill-the-capture incident.

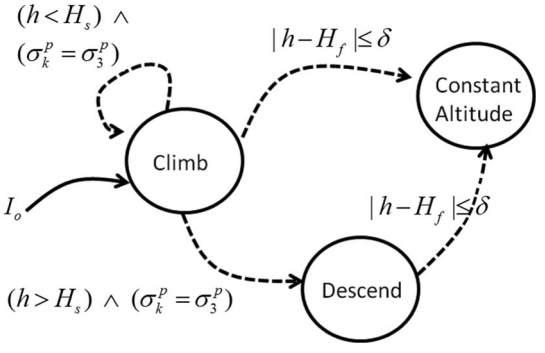

Fig. 7 Pilot's intent FSM for the kill-the-capture incident.

(i.e., pilot turns down the ALT knob and sets a lower target altitude value on the MCP) $\wedge\ \sigma_k^p :=$ (V/S button) is satisfied, it triggers an intent transition to the $I_{k+1}^p =$ descend state for the pilot. Similarly, other transitions can be understood. To run the algorithm, the value of $\delta = 100$ ft is considered.

d. Safety Specification: In the kill-the-capture incident, the mode confusion occurs when the pilot changes the MCP target altitude value from the previously set value of $H_f = 27{,}000$ ft to $H_s = 24{,}000$ ft while the autopilot is still in capture mode. If the capture start altitude H_c is between the newly set MCP target altitude H_s and the current altitude h_k, then the autopilot will autonomously enter the V/S climb mode and ignore the newly set altitude constraint, running contrary to the pilot's expectation/intent of descending aircraft motion. To detect this, we define

$$\bar{\phi} := \text{nodivergence} = \text{AG!}({}_V I_k^a = \text{climb} \ \&\ {}_V I_k^p = \text{descend}) \tag{15}$$

where & is the logical conjunction operator in the NuSMV.

e. Formal Model of the Pilot–Automation System and Model Checking: The aggregate pilot–automation interaction system is obtained by synchronously composing the intent-based FSMs of the automation (Fig. 6) and pilot (Fig. 7) by translating them to a single model in the Symbolic Model Verifier (SMV) language. When this aggregated formal model and the temporal logic safety specification (i.e., $\bar{\phi}$) are fed to the NuSMV model checker, the safety specification is monitored and a counterexample violating the specification is generated with detailed trace information (which helps to explain what led to this incident) in fraction of a second. We would like to note that this is possible because our method abstracts the infinite-dimensional hybrid model of the automation to a finite intent domain using predicate-based abstraction with qualitative reasoning, and the pilot is directly modeled in the finite intent domain, thus overcoming the state-space explosion problem that otherwise occurs when working directly in the hybrid domain. Specifically, $\bar{\phi} :=$ AG nodivergence is found to be false, meaning that a reachable state is detected in which the automation's intent is climb and the pilot's intent is "descend." This mismatch happens when the aircraft is climbing in the capture mode and the pilot turns down the ALT knob on the MCP. This results in an autonomous transition to the V/S climb (CLB) mode due to the satisfaction of the automation guard condition $H_s < H_c < h_k$, triggering a climb maneuver. However, for the pilot model, the corresponding guard condition is false, and hence does not match with that of the automation. Thus, the pilot's perceived behavior of the automation is inferred as the descend intent, which is in direct opposition to the automation's inferred climb intent. This causes violation of the safety specification (i.e., nodivergence = false), which causes a mode confusion. It can be easily seen that this incident happens due to a mismatch in the guard conditions between the automation and pilot models, and it is accurately detected by the proposed algorithm. Here, we must make the following remark: in this kill-the-capture incident, for the sake of simplicity and demonstration purposes, we have chosen to abstract away the speed and lateral dimensional states and focus only on the vertical dimensional behavior when analyzing for the mode confusion. Hence, the guard set D^a is set to true along the lateral and speed dimension intent transitions. Thus, for the detected counterexample, we have certainty about the detected confusion in the vertical dimension only (which is our particular interest in this incident), but not necessarily in the lateral and speed dimensions. What this means is that there exists a nondeterministic interpretation as to whether the aircraft is turning left or turning right when the mode confusion occurs due to our nature of modeling the behaviors in the vertical dimension only and disregarding the lateral and speed dimensional behaviors.

B. Example 2: Bangalore Accident

1. Description of Accident

The "Bangalore" accident of 1990 is a complex mode-confusion accident due to the involvement of both "autothrottle and autopilot" modes, as depicted in Fig. 8 [4,25]. On 14 February 14 1990, the crew of Indian Airlines Flight 605, which was an Airbus-320 aircraft, flying from Mumbai to Bangalore, India, upon approval at the control tower, decided to perform manual landing with the autothrottle on because the route was familiar. The aircraft began to capture the 3 deg glideslope using the V/S + speed (SPD) modes (autopilot and autothrottle modes, respectively) by performing a descent maneuver. During this period, the copilot mistakenly input a higher vertical speed, and the pilot input a target altitude lower than the current altitude by turning and pressing the ALT knob. This triggered an autonomous flight-mode transition to the open (OP) Descent (DES) + thrust (THR) idle modes, which went unnoticed by the crew (note that the open mode of Airbus is similar in functionality to the FLCH mode of Boeing, as shown in Table 1). If the pilot is manually flying in the OP mode, the pilot must follow flight director (FD) pitch guidance to maintain airspeed; otherwise, large airspeed deviations will occur. Belatedly realizing their mistake that the aircraft had descended too low with greatly reduced airspeed, but unaware of the cause, the captain hurriedly turned off his FD and increased the side stick input to full aft, expecting an autonomous flight-mode transition to the V/S + SPD modes to perform a climb maneuver and regain engine power to maintain the approach speed to keep the aircraft afloat. However, the internal logic of the automation worked such that the OP mode stayed active unless both the FDs were off. Since only the captain's FD was off, the autothrottle continued to provide idle thrust in the THR idle mode. The subsequent activation of alpha protection providing go-around thrust was too late, crashing the aircraft.

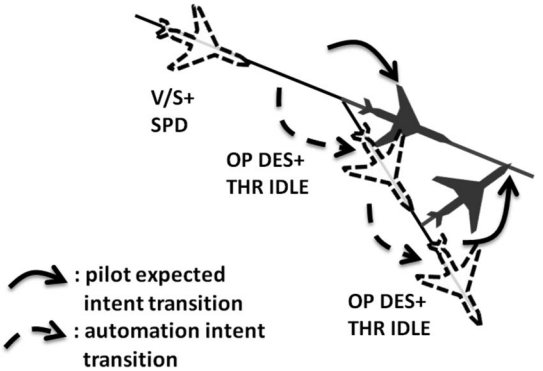

Fig. 8 Bangalore mode-confusion accident (actual trajectory given by aircraft with broken boundary and white interior, and pilot expected trajectory given by aircraft with solid boundary and gray interior).

2. Application of the Proposed Algorithm

We follow a similar analysis as in example 1 based on the proposed intent-based mode-confusion detection framework.

3. Formal Model of the Automation:

The hybrid system model for the automation for the Bangalore accident is discussed in the following. The mode-specific ($q \in Q = \{1, 2\}$) continuous dynamics $x_{k+1} = A_q x_k$, where $x_k = [h_k \dot{h}_k v_k \dot{v}_k]^T$ (h and $\dot{h}$ are the aircraft's altitude and altitude rate, respectively; and v and $\dot{v}$ are the aircraft's speed and speed rate, respectively) and A_q are 4×4 system matrices are given in the following [25]:

1) OP mode + THR idle mode, $q = 1$:

$$\begin{bmatrix} h_{k+1} \\ \dot{h}_{k+1} \\ v_{k+1} \\ \dot{v}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_s \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} h_k \\ \dot{h}_k \\ v_k \\ \dot{v}_k \end{bmatrix} \tag{16}$$

2) V/S mode + SPD mode, $q = 2$:

$$\begin{bmatrix} h_{k+1} \\ \dot{h}_{k+1} \\ v_{k+1} \\ \dot{v}_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & T_s & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} h_k \\ \dot{h}_k \\ v_k \\ \dot{v}_k \end{bmatrix} \tag{17}$$

The relevant flight-mode transitions are given by the automation's guard condition $D^a(\alpha, \beta), \forall\ \alpha, \beta \in Q = \{1, 2\}$ as (note here, again, that the autopilot can do more complex cases; but, to demonstrate the proposed method, a simple yet illustrative case is considered)

$$\begin{aligned} D^a(1,1) &= \{\sigma^p = \sigma_1^p\} \cup X \times \Sigma \setminus D^a(1,2) \\ D^a(1,2) &= \left\{ \left[[h\ \dot{h}\ v\ \dot{v}] \sigma^p \sigma^a \right]^T \middle| \left(\sigma^p = \sigma_5^p \vee (\sigma^p = \sigma_2^p \wedge u^p = u_1^p) \right) \right\} \\ D^a(2,1) &= \left\{ \left[[h\ \dot{h}\ v\ \dot{v}] \sigma^p \sigma^a \right]^T \middle| h > H_s > H_m \wedge \sigma^p = \sigma_3^p \right\} \\ D^a(2,2) &= X \times \Sigma \setminus D^a(2,1) \end{aligned} \tag{18}$$

In Eq. (18), superscript T refers to the transpose of a vector; $\sigma^p = \{\sigma_1^p, \sigma_2^p, \sigma_3^p, \sigma_4^p, \sigma_5^p, \sigma_6^p\} \in \Sigma^p$ where σ_1^p refers to the pilot's discrete control input of switching off one of the two FDs; σ_2^p refers to the pilot's discrete control input of switching off both the FDs; σ_3^p refers to the pilot pressing the ALT button; σ_4^p denotes the pilot turning the ALT knob to a newly set altitude H_s; $\sigma_5^p = \{\sigma_{5,1}^p, \sigma_{5,2}^p\}$, where $\sigma_{5,1}^p$ corresponds to the engagement of the appropriate autopilot mode (e.g., the V/S mode); $\sigma_{5,2}^p$ corresponds to the engagement of the appropriate autothrottle mode (e.g., the SPD mode); σ_6^p denotes the pilot turning the SPD knob to a newly set airspeed V_s; $u^p = \{u_1^p\} \in U^p$, where u_1^p refers to the pilot's side-stick aft input; and $\sigma^a = \{\sigma_{\text{auto}}^a\} \in \Sigma^a$ refers to the altitude h being greater than the missed approach altitude H_m.

4. Appropriate Flight Intent Set:

The intent state space for the Bangalore accident is given along the vertical and speed dimensions as

$$I \in \mathcal{I} = \{(\text{Climb, Accelerate}), (\text{Climb, Decelerate}), \cdots, (\text{Constant Altitude, Constant Speed})\} \tag{29}$$

Then, the intent-based FSM model of the automation can be obtained according to the intent transition logic in Eq. (5) for the preceding hybrid model, starting from the OP + THR idle mode and the (descend, decelerate) intent, and shown in Fig. 9. The following parameters are considered for initiating and running the algorithm [4]: $H_m = 600$ ft, $H_s = 700$ ft, $V_s = 132$ knot, initial altitude $h_0 = 1500$ ft, initial altitude rate $\dot{h}_0 = -15$ ft/s, initial airspeed $v_0 = 132$ knot, initial airspeed rate $\dot{v}_0 = 0$ kt/s, $\epsilon_r = 1$ ft/s, and $\epsilon_v = 1$ kt/s.

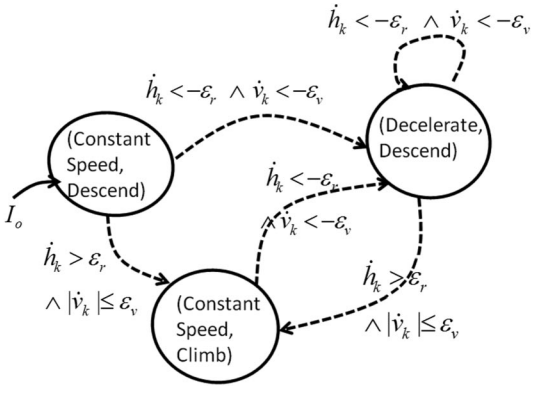

Fig. 9 Automation's intent FSM for the Bangalore incident.

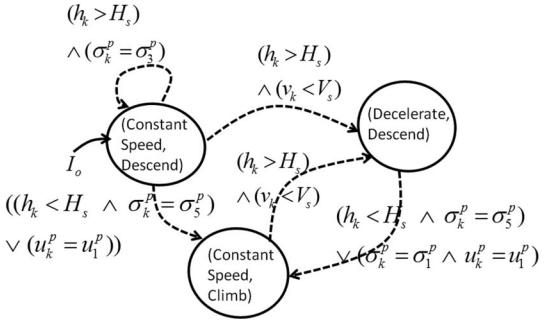

Fig. 10 Pilot's intent FSM for the Bangalore incident.

5. Formal Model of the Pilot:

For clarity, only the pilot's relevant perceived guard conditions $\bar{D}^p(i,j), \forall\ i,j = \{1,2,3\}$ [where 1 denotes (decelerate, descend), 2 denotes (constant speed, descend), and 3 denotes (constant speed, climb)] are given in the following; and the pilot's intent FSM is shown in Fig. 10 (only important transitions are indicated) as

$$
\begin{aligned}
\bar{D}^p(1,1) &= X \times \Sigma \setminus (\bar{D}^p(1,2) \cup \bar{D}^p(1,3))\\
\bar{D}^p(1,2) &= \left\{ \left[h\ \dot{h}\ v\ \dot{v} \right] \sigma^p \sigma^a \right]^T \middle| h > H_s \wedge \sigma^p = \sigma_5^p \right\}\\
\bar{D}^p(1,3) &= \left\{ \left[h\ \dot{h}\ v\ \dot{v} \right] \sigma^p \sigma^a \right]^T \middle| (h < H_s \wedge \sigma^p = \sigma_5^p) \vee (\sigma^p = \sigma_1^p \wedge u^p = u_1^p) \right\}\\
\bar{D}^p(2,1) &= \left\{ \left[h\ \dot{h}\ v\ \dot{v} \right] \sigma^p \sigma^a \right]^T \middle| h > H_s \wedge v < V_s \right\}\\
\bar{D}^p(2,2) &= (h > H_s \wedge \sigma^p = \sigma_3^p) \cup X \times \Sigma \setminus (\bar{D}^p(2,1) \cup \bar{D}^p(2,3))\\
\bar{D}^p(2,3) &= \left\{ \left[h\ \dot{h}\ v\ \dot{v} \right] \sigma^p \sigma^a \right]^T \middle| ((h < H_s \wedge \sigma^p = \sigma_{5,1}^p) \vee u^p = u_1^p) \right\}\\
\bar{D}^p(3,1) &= \left\{ \left[h\ \dot{h}\ v\ \dot{v} \right] \sigma^p \sigma^a \right]^T \middle| h > H_s \wedge v < V_s \right\}\\
\bar{D}^p(3,2) &= \left\{ \left[h\ \dot{h}\ v\ \dot{v} \right] \sigma^p \sigma^a \right]^T \middle| h > H_s \wedge \sigma^p = \sigma_{5,1}^p \right\}\\
\bar{D}^p(3,3) &= (h < H_s \wedge \sigma^p = \sigma_3^p) \cup X \times \Sigma \setminus (\bar{D}^p(3,1) \cup \bar{D}^p(3,2))
\end{aligned}
\tag{20}
$$

6. Safety Specification:

Based on the accident description at the beginning of this section, the safety specification in this mode-confusion accident is stated in ACTL using OR composition as $\bar{\Phi} = \{\bar{\phi}_1 | \bar{\phi}_2\}$, where

$$
\begin{aligned}
\bar{\phi}_1 &:= \text{AG nodivergence1} = !({}_V I_k^a = \text{descend} \ \&\ {}_V I_k^p = \text{descend} \ \&\ {}_S I_k^a = \text{decelerate} \ \&\ {}_S I_k^p = \text{constant speed})\\
\bar{\phi}_2 &:= \text{AG nodivergence2} = !({}_V I_k^a = \text{descend} \ \&\ {}_V I_k^p = \text{climb} \ \&\ {}_S I_k^a = \text{decelerate} \ \&\ {}_S I_k^p = \text{constant speed})
\end{aligned}
\tag{21}
$$

Here, $\bar{\Phi}$ means that, for all the exploratory paths of the model checker (and globally in the future), there exists no mismatch between the automation's and pilot's speed intent states as stated in nodivergence1; or, there exists no mismatch between their vertical and speed intent states as stated in nodivergence2. Note that the mode confusion in this accident is along both the vertical and speed dimensions, which is a more complex mode-confusion problem than the kill-the-capture incident; yet, the proposed intent-based framework works seamlessly with this case too.

7. Formal Model of the Pilot–Automation System and Model Checking:

The intent-based FSMs of the automation (Fig. 9) and pilot (Fig. 10) were synchronously composed by translating them in the SMV language, and the temporal logic safety specification given by $\bar{\Phi}$ was fed to the NuSMV model checker. It was found that a counterexample (i.e., $\bar{\phi}_1 =$ false or $\bar{\phi}_2 =$ false) was obtained along with detailed trace information in a fraction of a second. Specifically, $\bar{\phi}_1 =$ false, indicates a reachable state in which the automation's speed intent was "decelerate" and the pilot's speed intent was "constant speed", detecting the first instance of mode confusion in the Bangalore accident. This caused the aircraft speed to exceed the maximum airspeed in that configuration, triggering an autonomous mode transition. This resulted in the aircraft rapidly descending. Realizing this unexpected aircraft behavior, the pilot turned off one of the two FDs and pulled the side stick to full aft, expecting the aircraft to regain its speed and climb. However, the aircraft started to descend faster, resulting in $\bar{\phi}_2 =$ False. This means that a reachable state in which the automation's vertical intent was "descend" and the pilot's vertical intent is "climb", and the automation's speed intent is decelerate and the pilot's speed intent was constant speed was detected, indicating that both the automation's and the pilot's vertical and speed intents experienced a mismatch, causing the second instance of mode confusion in the Bangalore accident. This confirmed that the Bangalore accident, which comprised two sequentially occurring mode confusions, was correctly detected using the proposed intent-based mode-confusion detection framework. The detailed trace generated by the NuSMV model checker was omitted for brevity.

IV. Discussion and Future Work

Here, it is important to provide a comparative evaluation of the proposed method with the other common tools to deal with the complexity of formal verification of hybrid system analysis. These tools can be broadly classified into three main types (as noted by [37]): 1) tools that check for

correctness certificates such as barrier functions [44], 2) tools that compute an abstraction and then analyze the abstracted system [13,34,37,44–47], and 3) tools that overapproximate reach sets rather than constructing an abstraction [48]. The proposed abstraction framework falls in category 2, where different abstraction approaches exist to preserve different behaviors of the infinite-dimensional hybrid system. However, most existing tools often do not consider the property (e.g., mode confusion) itself when building an abstract model. Rather, an abstract model is initially constructed for the entire hybrid system using a certain degree of appropriate detail. If the abstraction is not sufficient to analyze the property, the whole abstraction process is started again, or the abstract model is globally refined [45]. Others have suggested a procedure that starts from a coarse model and a safety property, identifying parts of it that potentially violate the property and iteratively refining the model until verification reveals whether or not the property in question is satisfied [13,46]. But, unlike existing tools, in our work, the overall behavior of the hybrid system is abstracted with respect to sign-based predicates defined using qualitative reasoning of first-order continuous state rate information as suitable for mode-confusion detection (which is the property we are interested in evaluating). As noted in [47], the quality of the generated abstraction depends crucially on the choice of the predicates. The success of our approach depends on the chosen predicates for abstraction, which use flight intents. In particular, the behaviors inside a continuous evolution affected by the flight modes, control inputs, and ATC clearances are captured too. Such an intent-based abstraction allows us to obtain a finite state machine with discrete intent states for the automation's behavior and express the pilot's complex behavior as transitions between simple intents for computationally efficient detection of mode confusion.

However, there are a couple of interesting follow-up works to be explored. First, we note that deterministically inferring the intentions of the pilot may not be entirely accurate, especially when doing safety-critical verification for detection of unknown confusions. This is because the pilot's expected aircraft behavior (i.e., his/her intents) depends on the aircraft's continuous and discrete states, control actions, and knowledge of flight and training manuals. Each of the aforementioned factors contains uncertainties (e.g., navigation errors, control knob tuning), and the pilot's behaviors experience uncertainties in sensing, slips, lapses, variation in skills, and evolving expertise [17,28,39,40,49,50]. Therefore, it could be important to incorporate both continuous and discrete uncertainties in the autoflight system and uncertainties in the pilot behaviors in order to realistically characterize their interactions to capture nontrivial and (sometimes) unintended aircraft behaviors, as well as probabilistically infer their intents. The proposed mode-confusion detection framework can be extended to incorporate the effects of such uncertainties in both the aircraft's dynamics and the pilot's behaviors in order to quantify the likelihood of mode confusion and analyze the effects of varying the parameters of the pilot–automation formal model on mode confusion. An initial effort in this direction using probabilistic modeling techniques is in progress [42].

Second, additional work needs to be done to mechanize our algorithm:

1) A lot of human factor research [51–53] has attempted to incorporate erroneous human behaviors into the formal framework through temporal logic of actions (TLA), cognitive models, and operator function models; and how they contribute to system failures has been studies studied. Bolton and Bass [54] further proposed a method to automatically generate such erroneous human error behaviors and formally verified the human–automation interaction. However, not much research has been done to automatically generate erroneous pilot-intent models by incorporating pilot error templates for mode confusion, which can then be integrated it into our detection framework.

2) Because variables such as altitude, speed, etc., are generally of the real number type, it would be useful to automate the process of finding counterexamples to the desired specification of increasing length k (e.g., k-time steps) by using a Satisfiability Modulo Theories (SMT)-based infinite bounded model-checking tool (as provided by NuXMV, Symbolic Analysis Laboratory (SAL) tools). This tool would also allow the use of uninterpreted functions to code in the dynamics of real continuous variables for an automatic discrete-time update and return of a suitable predicate [13]. These aspects of automated analysis of our algorithm need to be explored in the near future.

Third, application of the proposed framework to more complex scenarios and to other human–machine systems for mode-confusion detection is possible. For example, confusions that occur due to time-varying aircraft behaviors such as faster descent, slower turn, etc., can be handled by augmenting the intent set with numeric parameters and broadening the current flight intent set using qualitative reasoning. Furthermore, complex intents can be made by a sequence of simple intents; for example, an aircraft flying to a desired waypoint located at a higher altitude (a case of vertical maneuver), where the complex intent could be "go-to-waypointB," can be represented by a sequence of tactical intents: "climb with constant speed and constant heading" → "capture altitude and constant heading" → "constant altitude at constant speed and constant heading." Similarly, other complex intents can be defined such as {hold-pressure-altitude, speed-to-meet-Miles-in-trail (MIT)-restriction, capture airway} [36]. Then, a strategic level confusion, if it exists, can be detected as a mismatch between the pilot's and the automation's tactical intent sequences. Finally, the intent concept is general enough that it could be applied to various applications such as robot–human systems, autonomous cars, and unmanned aircraft systems (e.g., drones). For example, in a robot manipulator–human system application, an intent set could be either {pick-up-an-object, grasp-an-object, drop-an-object, place-an-object} or {increase-distance-to-object, decrease-angle-to-object, go-to-waypoint1, hold-the-current-course}, depending on the scenario or level at which we are interested to address the mode confusion, where the human/robot achieves these goals via appropriate interactions.

Fourth, in this paper, we have not explored ways to fix the detection confusions. Since one of the main causes of mode confusion is the inadequate representation of information on the user interface [42], one way to address this is to introduce a UI model into the formal pilot–automation model (that we have currently discussed in this paper) to serve as a (binary) filter (i.e., choose to either display or not display certain flight information such as flight modes). We anticipate this introduction of UI and its synchronous composition with the pilot–automation model would enable even better characterization of the pilot–automation interaction for mode-confusion detection and help in the counterexample guided abstraction refinement of the UI model to fix and eliminate the detected mode confusion without potentially introducing newer confusions.

V. Conclusions

This paper has proposed a computationally efficient formal verification framework to detect a wide range of mode-confusion problems in the complex pilot–automation system that occur due to both discrete and continuous deviations of aircraft that run contrary to the pilot's expectations. The framework was further demonstrated with two real mode-confusion incidents/accidents. The proposed approach constructed modular, interacting formal state-space models for the automation as a hybrid system to capture the continuous and discrete behaviors of the aircraft, as well as the pilot as an intent-based finite state machine to describe his/her decision making. To facilitate computationally feasible verification and further enable direct comparison of the otherwise heterogeneous states of the automation and pilot, an intent-based abstraction was proposed to obtain an intent-based FSM model for the automation. The proposed mode-confusion detection framework was intended to complement and augment the existing works in human factors, data mining, and simulation for systematic analysis of a wide range of mode-confusion problems.

Acknowledgments

The authors would like to acknowledge that this work is supported by NSF CMMI 1335084, and they thank the reviewers for their useful comments.

References

[1] Degani, A., Heymann, M., Meyer, G., and Shafto, M., "Some Formal Aspects of Human-Automation Interaction," NASA Ames Research Center TM 2000-209600, Moffett Field, CA, 2000.
[2] Crow, J., Javaux, D., and Rushby, J., "Models and Mechanized Methods that Integrate Human Factors into Automation Design," *International Conference on Human-Computer Interaction in Aeronautics: HCI-Aero*, Tolosa Press, Toulouse, France, 2000, pp. 163–168.
[3] "Final Report of the Accident Investigation Flash Airlines Flight 604, Boeing 737-300, SU-ZCF, January 3, 2004, Red Sea near Sharm El-Sheikh, Egypt," Egyptian Ministry of Civil Aviation, Final Investigation Report, Cairo, Egypt, 2004.
[4] "Report on Accident to Indian Airlines Airbus A-320 Aircraft VT-EPN at Bangalore on 14th February 1990," Indian Court of Inquiry, Indian Government, Final Investigation Report, New Delhi, India, 1992.
[5] Sarter, N., and Woods, D. D., "Decomposing Automation: Autonomy, Authority, Observability and Perceived Animacy," *1st Automation Technology and Human Performance Conference*, Erlbaum, Hillsdale, NJ, 1994, pp. 22–26.
[6] Sheridan, T. B., and Parasuraman, R., "Human-Automation Interaction," *Reviews of Human Factors and Ergonomics*, Vol. 1, No. 1, 2005, pp. 89–129. doi:10.1518/155723405783703082
[7] Miller, S. P., and Potts, J. N., "Detecting Mode Confusion Through Formal Modeling and Analysis," NASA Langley Research Center Rept. NASA/CR-1999-108971, Hampton, VA, 1999.
[8] Masci, P., Curzon, P., Blandford, A., and Furniss, D., "Modeling Distributed Cognition Systems in PVS," *Proceedings of the 4th International Workshop on Formal Methods for Interactive Systems*, Vol. 45, Electronic Communications of the EASST (ECEASST), TU Berlin, 2011.
[9] Das, S., Matthews, B. L., Srivastava, A. N., and Oza, N. C., "Multiple Kernel Learning for Heterogeneous Anomaly Detection: Algorithm and Aviation Safety Case Study," *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, Washington, D.C., 2010, pp. 47–56.
[10] Li, L., Gariel, M., Hansman, R. J., and Palacios, R., "Anomaly Detection in Onboard-Recorded Flight Data Using Cluster Analysis," *Proceedings of the 30th Digital Avionics Systems Conference (DASC)*, IEEE Publ., Piscataway, NJ, 2011, Paper 4A4-1.
[11] Butler, R. W., Miller, S. P., Potts, J. N., and Carreno, V. A., "A Formal Methods Approach to the Analysis of Mode Confusion," *17th DASC Proceedings for Digital Avionics Systems Conference*, Vol. 1, IEEE Publ., Piscataway, NJ, 1998, Paper C41-1.
[12] Rushby, J., "Analyzing Cockpit Interfaces Using Formal Methods," *Electronic Notes in Theoretical Computer Science*, Vol. 43, May 2001, pp. 1–0. doi:10.1016/S1571-0661(04)80891-0
[13] Bass, E. J., Feigh, K. M., Gunter, E., and Rushby, J., "Formal Modeling and Analysis for Interactive Hybrid Systems," *4th International Workshop on Formal Methods for Interactive Systems: FMIS*, Vol. 45, Electronic Communications of the EASST, Electronic Communications of the EASST, TU Berlin, 2011.
[14] Degani, A., "Modeling Human-Automation Interaction Using Finite State Machines Formalism," *Proceedings of the Workshop on Formal Methods in Human-Machine Interaction*, Vol. 45, IFIP Working Group 13.5 on Human Error, Safety, and System Development, Imperial College, London, 2011, pp 37–40.
[15] Bolton, M. L., Bass, E. J., and Siminiceanu, R. I., "Using Formal Verification to Evaluate Human-Automation Interaction: A Review," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, Vol. 43, No. 3, March 2013, pp. 488–503. doi:10.1109/TSMCA.2012.2210406
[16] Hu, A. J., "Simulation vs. Formal: Absorb What Is Useful; Reject What Is Useless," *Haifa Verification Conference*, Springer, New York, 2007, pp. 1–7.
[17] Javaux, D., "A Method for Predicting Errors When Interacting with Finite State Systems. How Implicit Learning Shapes the User's Knowledge of a System," *Reliability Engineering and System Safety*, Vol. 75, No. 2, 2002, pp. 147–165. doi:10.1016/S0951-8320(01)00091-6
[18] Oishi, M., Mitchell, I., Bayen, A., and Tomlin, C., "Hybrid System Verification: Application to User Interface Design," *IEEE Transactions on Control Systems Technology*, Vol. 16, No. 2, 2003, pp. 229–244. 10.1109/TCST.2007.903370
[19] Oishi, M., Mitchell, I., Bayen, A., Tomlin, C., and Degani, A., "Hybrid Verification of an Interface for an Automatic Landing," *Proceedings of the 41st IEEE Conference on Decision and Control (CDC)*, Vol. 2, IEEE Publ., Piscataway, NJ, 2002, pp. 1607–1613.
[20] Sherry, L., Feary, M., Polson, P., and Palmer, E., "Formal Method for Identifying Two Types of Automation-Surprises," Honeywell TR C69-5370-016, Phoenix, AZ, 2000.
[21] Mitchell, C. M., and Miller, R. A., "A Discrete Control Model of Operator Function: A Methodology for Information Display Design," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 16, No. 3, 1986, pp. 343–357. doi:10.1109/TSMC.1986.4308966
[22] Paternò, F., Mancini, C., and Meniconi, S., "ConcurTaskTrees: A Diagrammatic Notation for Specifying Task Models," *Human-Computer Interaction INTERACT97*, Springer, New York, 1997, pp. 362–369.
[23] Bolton, M. L., and Bass, E. J., "Using Model Checking to Explore Checklist-Guided Pilot Behavior," *International Journal of Aviation Psychology*, Vol. 22, No. 4, 2012, pp. 343–366. doi:10.1080/10508414.2012.718240
[24] Bolton, M. L., "Automatic Validation and Failure Diagnosis of Human-Device Interfaces Using Task Analytic Models and Model Checking," *Computational and Mathematical Organization Theory*, Vol. 19, No. 3, 2013, pp. 288–312. doi:10.1007/s10588-012-9138-6
[25] Nandiganahalli, J. S., Lee, S., Hwang, I., and Yang, B.-J., "User Interface Validation Using Mode Confusion Detection," *AIAA Modeling and Simulation Technologies Conference*, AIAA Paper 2014-2349, 2014.
[26] Lee, S., Hwang, I., and Leiden, K., "Intent Inference-Based Flight-Deck Human-Automation Mode-Confusion Detection," *Journal of Aerospace Information Systems*, Vol. 12, No. 8, 2015, pp. 503–518. doi:10.2514/1.I010331
[27] Nandiganahalli, J. S., Lyu, H., and Hwang, I., "Formal Extensions to the Intent-Based Mode Confusion Detection Framework," *AIAA Modeling and Simulation Technologies Conference*, AIAA Paper 2015-2335, 2015.
[28] Landry, S. J., *Advances in Human Aspects of Aviation*, CRC Press, Boca Raton, FL, 2012, Chaps. 6–8.
[29] Tomlin, C. J., Mitchell, I., Bayen, A. M., and Oishi, M., "Computational Techniques for the Verification of Hybrid Systems," *Proceedings of the IEEE*, Vol. 91, No. 7, 2003, pp. 986–1001.
[30] Mitchell, I., and Tomlin, C. J., "Level Set Methods for Computation in Hybrid Systems," *International Workshop on Hybrid Systems: Computation and Control*, Lectures in Computer Science, Springer, New York, 2000, pp. 310–323.
[31] Stipanović, D. M., Hwang, I., and Tomlin, C. J., "Computation of an Over-Approximation of the Backward Reachable Set Using Subsystem Level Set Functions," *European Control Conference (ECC), 2003*, IEEE Publ., Piscataway, NJ, 2003, pp. 300–305.
[32] Hwang, I., Stipanovic, D. M., and Tomlin, C. J., "Applications of Polytopic Approximations of Reachable Sets to Linear Dynamic Games and a Class of Nonlinear Systems," *Proceedings of the 2003 American Control Conference*, Vol. 6, IEEE Publ., Piscataway, NJ, 2003, pp. 4613–4619.
[33] Hwang, I., Balakrishnan, H., Ghosh, R., and Tomlin, C., "Reachability Analysis of Delta-Notch Lateral Inhibition Using Predicate Abstraction," *International Conference on High-Performance Computing*, Springer, New York, 2002, pp. 715–724.

[34] Sankaranarayanan, S., and Tiwari, A., "Relational Abstractions for Continuous and Hybrid Systems," *International Conference on Computer Aided Verification*, Springer, New York, 2011, pp. 686–702.
[35] Cimatti, A., Clarke, E., Giunchiglia, F., and Roveri, M., "NuSMV: A New Symbolic Model Verifier," *International Conference on Computer Aided Verification*, Springer, New York, 1999, pp. 495–499.
[36] Yepes, J. L., Hwang, I., and Rotea, M., "New Algorithms for Aircraft Intent Inference and Trajectory Prediction," *Journal of Guidance, Control, and Dynamics*, Vol. 30, No. 2, 2007, pp. 370–382. doi:10.2514/1.26750
[37] Tiwari, A., "Abstractions for Hybrid Systems," *Formal Methods in System Design*, Vol. 32, No. 1, 2008, pp. 57–83. doi:10.1007/s10703-007-0044-3
[38] Loeser, T., Iwasaki, Y., and Fikes, R., "Safety Verification Proofs for Physical Systems," *Proceedings of the 12th International Workshop on Qualitative Reasoning*, AAAI Press, Cambridge, MA, 1998, pp. 88–95.
[39] Seah, C. E., and Hwang, I., "Stochastic Linear Hybrid Systems: Modeling, Estimation, and Application in Air Traffic Control," *IEEE Transactions on Control Systems Technology*, Vol. 17, No. 3, 2009, pp. 563–575. doi:10.1109/TCST.2008.2001377
[40] Zhao, Y. J., and Zheng, Q. M., "Modeling Uncertainties in Intents, Guidance, and Pilot Actions for Advanced Trajectory Synthesis," *12th AIAA Aviation Technology, Integration, and Operations (ATIO) Conference and 14th AIAA/ISSM*, AIAA Paper 2012-5421, 2012.
[41] Casner, S. M., "Understanding the Determinants of Problem-Solving Behavior in a Complex Environment," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, Vol. 36, No. 4, 1994, pp. 580–596.
[42] Nandiganahalli, J. S., Lee, S., and Hwang, I., "Flight Deck Mode Confusion Detection Using Intent-Based Probabilistic Model Checking," *AIAA SciTech 2017: AIAA Infotech@Aerospace* (accepted for publication).
[43] Meolic, R., Kapus, T., and Brezocnik, Z., "Verification of Concurrent Systems Using ACTL," *Proceedings of the IASTED International Conference on AI*, Applied Informatics, Springer, New York, 2000, pp. 663–669.
[44] Prajna, S., and Jadbabaie, A., "Safety Verification of Hybrid Systems Using Barrier Certificates," *International Workshop on Hybrid Systems: Computation and Control*, Springer, New York, 2004, pp. 477–492.
[45] Chutinan, A., and Krogh, B. H., "Verification of Infinite-State Dynamic Systems Using Approximate Quotient Transition Systems," *IEEE Transactions on Automatic Control*, Vol. 46, No. 9, 2001, pp. 1401–1410. doi:10.1109/9.948467
[46] Clarke, E., Fehnker, A., Han, Z., Krogh, B., Stursberg, O., and Theobald, M., "Verification of Hybrid Systems Based on Counterexample-Guided Abstraction Refinement," *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Springer, New York, 2003, pp. 192–207.
[47] Alur, R., Dang, T., and Ivančić, F., "Counter-Example Guided Predicate Abstraction of Hybrid Systems," *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Springer, New York, 2003, pp. 208–223.
[48] Dang, T., "Approximate Reachability Computation for Polynomial Systems," *International Workshop on Hybrid Systems: Computation and Control*, Springer, New York, 2006, pp. 138–152.
[49] Reason, J., "Human Error: Models and Management," *BMJ*, Vol. 320, No. 7237, 2000, pp. 768–770. doi:10.1136/bmj.320.7237.768
[50] Hollnagel, E., "The Phenotype of Erroneous Actions," *International Journal of Man-Machine Studies*, Vol. 39, No. 1, 1993, pp. 1–32. doi:10.1006/imms.1993.1051
[51] Johnson, C. W., and Telford, A. J., "Extending the Application of Formal Methods to Analyse Human Error and System Failure During Accident Investigations," *Software Engineering Journal*, Vol. 11, No. 6, 1996, pp. 355–365. doi:10.1049/sej.1996.0046
[52] Lindsay, P., and Connelly, S., "Modelling Erroneous Operator Behaviours for an Air-Traffic Control Task," *Australian Computer Science Communications*, Vol. 24, Australian Computer Soc., Sydney, 2002, pp. 43–54.
[53] Bolton, M. L., and Bass, E. J., "Formal Modeling of Erroneous Human Behavior and Its Implications for Model Checking," *Proceedings of 6th NASA Langley Formal Methods Workshop*, NASA Langley Research Center CP 2008-215309, Hampton, VA, 2008, pp. 62–64.
[54] Bolton, M. L., and Bass, E. J., "Evaluating Human-Automation Interaction Using Task Analytic Behavior Models, Strategic Knowledge-Based Erroneous Human Behavior Generation, and Model Checking," *IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, IEEE Publ., Piscataway, NJ, 2011, pp. 1788–1794.

M. D. Davies
Associate Editor