ICAS 2016
30th Congress of the International Council of the Aeronautical Sciences
DCC, Daejeon, Korea ; September 25-30, 2016

FORMAL VERIFICATION OF AUTOPILOTS IN UNMANNED AIRCRAFT SYSTEMS

Bong-Jun Yang*, Parikshit Dutta*, Insoek Hwang**
***Optimal Synthesis Inc., ****School of Aeronautics and Astronautics, Purdue University**

Keywords: *Model Checking, Formal Verification, UAV Autopilot, Verification and Validation*

Abstract

The verification and validation (V&V) and certification problems for avionics systems in Unmanned Aircraft Systems (UAS) have been regarded as great challenges in realizing routine UAS into the National Air Space (NAS). Whereas current-day emerging avionics systems, such as embedded autopilots in UAS, are geared towards more autonomous operations, the currently employed V&V techniques in the industry largely rely on exhaustive testing, and this poses a technology gap between the software capability and the certification techniques in that airworthiness of autonomous systems is limited by the only certifiable operations, i.e., fixed, pre-planned operations considered in test scenarios.

The research effort in this paper attempts to reduce the technology gap by developing a V&V technique that formally verifies a flight system instead of accumulating confidence by test cases. The approach relies on the promise demonstrated by the model checking technique in formally verifying the safety and correctness of finite-state machine systems and tries to extend its validity into the verification of UAS autopilot systems. The paper includes the description of applied model checking methods, underlying algorithms that abstract the hybrid system into a finite-state machine, and simulation results.

1 Introduction

Demanded by the growing use of unmanned aircraft systems (UAS) in commercial, governmental, and military sectors, the integration of UAS into the National Air Space (NAS) has become a national interest. In response to the demand for the routine access of UAS to the NAS, the Next Generation Air Transportation System (NextGen) partner agencies and industry representatives have created a comprehensive plan under the vision:"UAS must be integrated into the NAS without reducing existing capacity, decreasing safety, negatively impacting current operators, or increasing the risk to airspace users of persons or property on the ground any more than the integration of comparable new and novel technologies." [1] Along with the requirement, National Aeronautics and Space Administration (NASA) has also put one of its strategic thrust on "Assured Autonomy of Aviation Transformation," in which a great emphasis goes to the safety as one of properties that must be assured in the effort of integrating UAS into the NAS. However, the statistics on the UAS mishaps record indicate that the mishap rate in UAS operations as of 2005 is orders-of-magnitude higher than that in manned aviation [2]. This issue has raised a public concern on the safety and reliability of UAS operations in the NAS. Among the technical barriers associated with the safety and the reliability, verification and validation (V&V) and certification problems for UAS operations are widely regarded as a stumbling block for routine UAS access to the NAS.

The difficulty in V&V and certification for the safety of UAS operations is attributable to the unproved and unforeseen nature of UAS operations, dictated by their autonomous mission planning and flight control systems in dynamically changing environment. From the perspective of autonomous operations, Unmanned Aerial Vehicles (UAVs) are desired to be able to react to unplanned events,

however, current test-based V&V approaches cannot handle autonomous operations because all possible outcomes of autonomous operations cannot be well captured by prediction-based test scenarios. If all potential events cannot be predicted, then there will be lack of contingency plans, which need to be tested, for certain missing events [3]. Moreover, it is anticipated that as the algorithms become progressively complex, the test-based V&V methods will become prohibitively costly and ultimately infeasible at achieving safety confidence [4].

The approach employed in this paper builds on the potential of formal verification methods demonstrated in the software verification domain [5]. The formal methods have generally been shown to be superior to manual testing and particularly suitable for finding certain class of errors [6]. The formal methods have also been accepted as a supplementary method for DO-178C "Software Considerations in Airborne Systems and Equipment Certification" by DO-333 "Formal Methods Supplement to DO-178C and DO-278A," which becomes a basis for the model checking [7] method as an alternative verification method to the test-based verification method.

In this paper, we apply the model checking method to the verification of an autopilot system employed in UAS. One immediate issue in applying model checking to UAV autopilots is the fact that model checking can only be applied to finite-state machine (FSM) whose states are completely discrete states. UAV flight control systems are generally described as a hybrid system in which discrete dynamics, such as various autopilot modes that determine the flight phase and logical features of UAV operations, and continuous dynamics, such as the time evolution of the air speed, the altitude, and the attitude of UAV, are tightly coupled. This means that for the UAV autopilots to be verified by model checking there must a mechanism that extracts a FSM from the original hybrid system description while the logical property of certain features, for example, safety, in the extracted FSM guarantees the property of the original hybrid system. We recognize that reachable set decomposition that abstracts the hybrid system into a FSM provides such a guaranteed framework for hybrid systems with state-dependent transitions among discrete states [8]. In other words, the reachable set decomposition combined with model checking allows the model-checking tool to verify the safety property of the original, hybrid UAV flight control systems. Since the model checking is performed by a computational tool, this process can reduce the cost associated with iterative testing by replacing the process with the mathematical guarantee for the safety.

2 Technical Approach

Fig. 1 depicts the process of model checking for UAV autopilots, which is described as a hybrid system. The verification is done using reachability analysis, where we calculate the reachable sets for the UAV states given different autopilot modes and initial state sets. The safety property is then specified as a set condition; if the state enters the set restricted by the safety consideration, such as outside of a flight envelope, the safety deems violated, otherwise, the system is viewed to remain safe. If the reachability analysis fails to assure the safety, it means that the reachable set overlaps with the unsafe set. In such a case, the reachability computation can be performed again after the approximation is further refined. If no refinement is possible then the system would deem unsafe.

When the system is deemed unsafe, it is desired to obtain a trajectory that indeed violates the safety property. In exact hybrid-system reachability computation, finding a counter example naturally comes out of reachability set computation [9]. However, in approximate reachability-based method, the overlap of the reachable set with the unsafe set does not necessarily mean the existence of a safety-violating trajectory in the original system. In such a case, the refinement (by reducing approximation error) is combined with search

for a trajectory that violates the safety property. In other words, in approximate model checking, the reachability analysis only provides a sufficient condition for the system safety, not a necessary condition. In our effort, we have applied reachability-based formal verification of UAV autopilot using the SpaceEx tool [10].

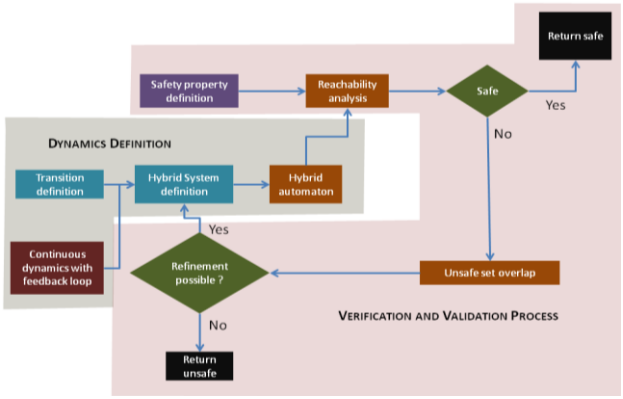


Fig. 1 Forma verification process.

2.1 Rechability Analysis

Let us consider the following dynamical system,

$$\frac{dx(t)}{dt} = f(x(t), t), x(0) \in X_0 \qquad (1)$$

where $x \in R^n$ is the state vector, X_0 describes the set of initial states and $t \in R^+$ is the time. $f: R^n \times R^+ \mapsto R^n$ is the function vector describing the dynamics. The set of possible states at any time t, can be mapped from the initial states X_0 of the system. These sets are called reachable sets. An example of the reachable sets for a given initial state set can be seen in Fig. 2. In general, the reachable sets can be arbitrary and sometimes may be disjoint. Hence, it is difficult to exactly represent the reachable sets.

This problem of exact reachable set computation can be circumvented by using polyhedral approximation of the reachable set. The reachable sets are represented as a convex polyhedron over the actual sets at a given time.

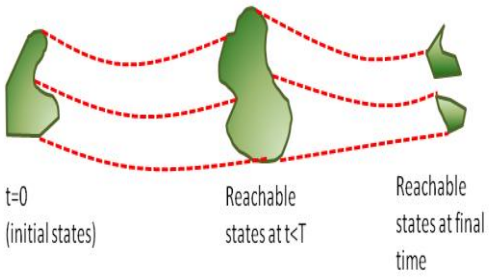


Fig. 2 An example of reachable sets.

In a way it overapproximates the convex hull of the sets to account for unknown disturbances. For a given period of time $[0, T]$, the reachable polyhedrons are combined to form what is called a *flow pipe* as depicted in Fig. 3. In the PHAVer tool [11], the flow pipe is computed by propagating the discretized version of the linear system forward in time to obtain reachable sets. A linear transformation is used to recursively calculate the reachable polyhedron, the vertices of which are computed using Minkowski sum. One drawback of PHAVer is that, the accuracy of the reachable sets thus created depends on how the noise is overapproximated and can often lead to conservative estimates.

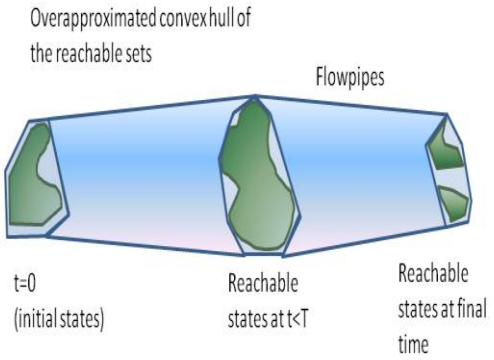


Fig. 3. Polyhedral approximation for flow pipes.

The conservatism associated with overapproximation is alleviated by the Le–Guernic Girard (LGG) algorithm, which uses support functions to calculate reachable sets. The support functions relax the assumption of polyhedrons and allow for including zonotopes, ellipsoids, and polytopes for reachable set computation. Therefore, it can make

overapproximation error arbitrarily small, which also requires employing a sufficiently small time step in system propagation. A support function of a set Ω is given by,

$$\rho_{\Omega}(l) \coloneqq \max_{x \in \Omega} l^T x$$

For example in Fig. 4, the support of a convex set has been illustrated. Once the support functions have been calculated for an arbitrary convex set, it can be propagated forward in time to obtain reachable sets. Details of how support functions are used to calculate reachable sets are found in Ref. [12].

2.2 Hybrid Input-Output Automata with Affine Dynamics

In this research, we have used the simple concept of input-output automata, which is largely based on hybrid automata described in Ref. [13]. A hybrid automaton can be described as a FSM with a finite set of continuous variables whose values are described by a set of ordinary differential equations. The dynamics describing the FSM changes when some guard conditions on the current state are met. Complete mathematical description of the hybrid automaton can be found in Ref. [13]. The dynamics of system should be represented as a linear combination of states plus a constant term. In other words, given the system state $x \in R^n$, $\dot{x} = Ax + b, A \in R^{n \times n}, b \in R^n$. If all the dynamics are given by the linear transformation as described above we call it affine hybrid dynamics.

This work calculates the reachable states of a system having hybrid affine dynamics, which is then further used to verify the dynamical system.

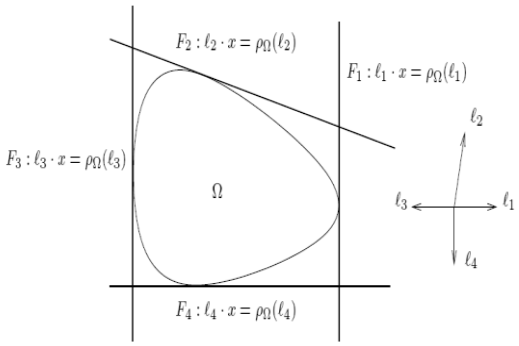


Fig. 4 Illustration of the notion of support functions [12]

2.3 Verification of Affine Hybrid Systems

Once the hybrid dynamics has been determined, and the hybrid automaton is defined, it is necessary to define a safety property for verification. Safety property can be represented as algebraic function/functions of states. For every time instant t, it is verified whether the reachable sets violate the safety property defined. In case the safety property is violated, the system is deemed unsafe.

Fig. 5 describes the methodology of verification employed for a single-state system. The unsafe region (marked by red) is described as an inequality given by, $a \leq x \leq b$. The blue bars at time t_1, t_2 and t_3 represents reachable sets for systems 1 and 2, given the initial state set at $t = 0$. It can be seen that the system 1 reaches the unsafe region and hence is unsafe whereas system 2 stays away from the unsafe region and is safe.

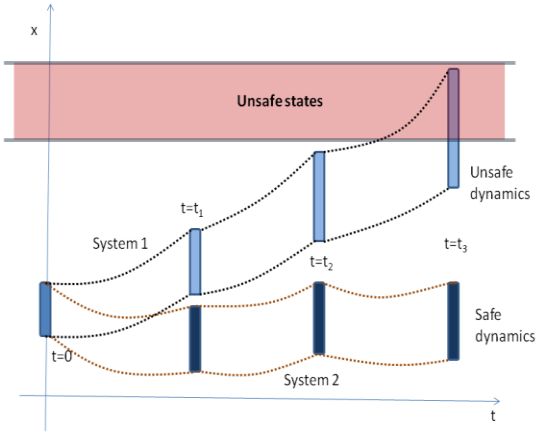


Fig. 5 Reachability-based verification.

3 Verification of a small UAV autopilot

3.1 UAV Dynamics

UAV autopilot verification in the longitudinal channel is considered for feasibility study. Under the assumption that the airspeed and the flight path angle are well regulated, the UAV dynamics are governed by:

$$\dot{x} = V \cdot \cos\gamma, \tag{2}$$

$$\dot{V} = -K_v(V - V_c), \tag{3}$$

$$\dot{h} = V \cdot \sin\gamma, \tag{4}$$

$$\dot{\gamma} = -K_\gamma(\gamma - \gamma_c) \tag{5}$$

where x, V, h and γ are the position, airspeed, altitude and flight path angle of the UAV, respectively. Here position refers to the position along x-axis which is along the surface of the earth. K_v and K_γ are time-constant achieved by the airspeed and the flight path angle control system. The variables V_c and γ_c denote the commanded airspeed and commanded flight path angle.

Since the flight path angle is usually small, the corresponding linearized model is given by the following equations:

$$\dot{x} = V, \tag{6}$$

$$\dot{V} = -K_v(V - V_c), \tag{7}$$

$$\dot{h} = V \cdot \gamma, \tag{8}$$

$$\dot{\gamma} = -K_\gamma(\gamma - \gamma_c) \tag{9}$$

Fig. 6 shows the altitude time responses of the nonlinear and linear models when the commanded flight path angle varies with $\pm 7^0$. They are almost identical, implying the linear model is a close approximation to the nonlinear system.

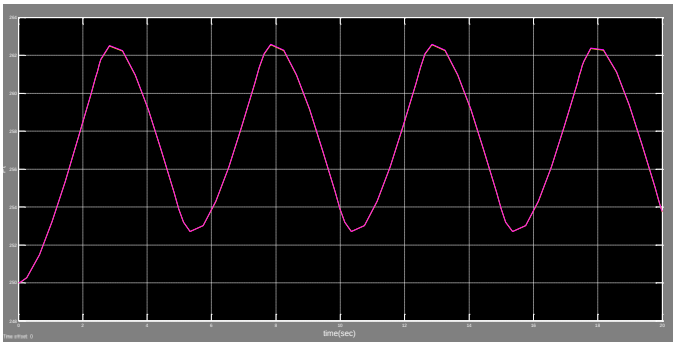


Fig. 6 Comparison of altitude time responses for linear and nonlinear modes

3.2 Verification Scenario

For verification of the small UAV autopilot, we use a scenario as described in Fig. 7.

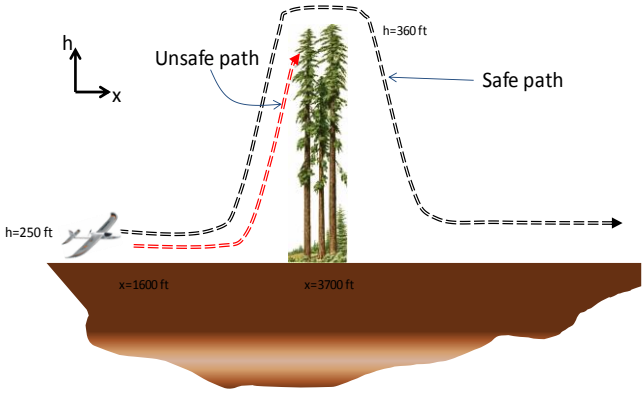


Fig. 7 Scenario used for verification of autopilot logic.

We let the UAV start from x=1600 ft and around x=3700 ft there is a collection of redwood trees. The redwood trees are assumed to cover a circular area with 100 ft radius. Moreover the height of the tallest redwood tree is 360 ft. Initially the UAV is assumed to cruise at x=0, h=250ft with v=30 knots. We need to verify if the autopilot described in the next section does not violate the safety constraints which are hitting the redwood trees.

3.3 Altitude Autopilot Design

An autopilot is assumed to be designed such that the UAV changes the flight path angle in order to regulate the altitude while maintaining the constant airspeed. The UAV is assumed to be limited in the engine, and therefore the vertical rate for climb is limited. When the flight

path angle is treated as a control signal, this implies that the flight path angle is limited. More specifically, $\underline{\gamma}(=-8^0) \leq \gamma \leq \overline{\gamma}(=7^o)$ leads to $\underline{\dot{h}}(=V_c \sin(\underline{\gamma}) \leq \dot{h} \leq \overline{\dot{h}}(=V_c \sin(\overline{\gamma}))$ for maximum design descent rate and the maximum design climb rate, respectively.

While some autopilots consider the altitude and the airspeed by considering the energy for smart use of engines, the rationale in this section for the altitude regulation resembles that of the commercial airlines. When the target altitude is far away from the current altitude, the vertical speed is set to either $\overline{\dot{h}}$ for climb or $\underline{\dot{h}}$ for descent. Otherwise (if the target altitude is close), the target altitude is tracked by altitude regulation control system. Conceptually, these are analogous to the vertical speed (V/S) mode and altitude hold (ALT HLD) mode in commercial airplanes. In other words, the commanded flight path angle γ_c is determined by:

$$\gamma_c = \begin{cases} k_p(h_c - h) & if \quad |h_c - h| \leq \varepsilon_h \\ \overline{\gamma} & if \quad climb\ and\ |h_c - h| > \varepsilon_h \\ \underline{\gamma} & if \quad \overline{\gamma} \quad if \quad descent\ and\ |h_c - h| > \varepsilon_h \end{cases} \tag{10}$$

For the current task the parameters $K_v = 0.5$ and $K_\gamma = 1.5$. $V_c = 30$ knots and is kept constant throughout the mission. The control gain k_p=0.0135 is selected such that the bandwidth of the closed-loop altitude hold mode is approximately one third of the bandwidth of the airspeed loop. As can be seen in Fig. 8, a single proportional gain is sufficient to achieve a well-performing altitude control system.

In the setting of avoiding redwood trees, the above autopilot logics lead to mode transitions depicted in Fig. 9. In order to avoid the trees, the mission profile of the altitude changes from the initial 250ft to 400 ft in order to avoid the tree and then is set back to 250ft to recover the initial altitude.

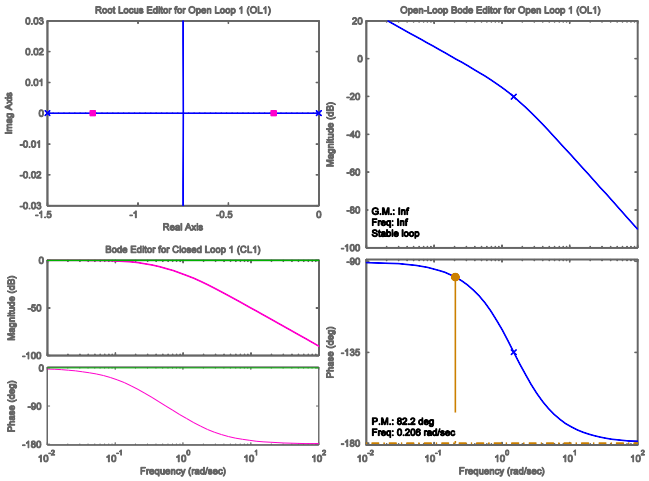


Fig. 8 Design diagram for the altitude loop using MATLAB SISOTOOL

The switch of autopilot logic from ALT HLD to V/S Climb and V/S Descent converts the closed loop structure of the UAV into an affine hybrid system. Hence techniques described in the foregoing sections can be used to formally verify the safety of the UAV against the scenario. The switching from ALT HLD mode to V/S Climb mode occurs when the UAV reaches x = 2600 ft, further switching from the V/S Climb mode to the ALT HLD mode is carried out at h=380ft with $\varepsilon_h = 20\ ft$. Further the ALT HLD mode to the V/S Decent mode transition happens at x = 3800 ft and from the V/S Descent mode to the ALT HLD mode at h=270 ft.

3.4 Results of Verification

Next we verify if the autopilot logic described in the previous section is safe to carry out the mission. The initial states are varied between different ranges. It is checked if the autopilot logic yields safe path for all values of initial cases.

First we vary the initial starting position along x axis from 1600 to 2000 ft. That is 1600 ft $\leq$ x($t = 0$) $\leq$ 2000 ft. Then along with x, initial altitude (h) is varied from 164 ft $\leq$ h($t = 0$) $\leq$ 250 ft. Further we vary x, h and v, where 23 knots $\leq$ v($t = 0$) $\leq$ 30 knots. Finally all initial states are varied where,

$0^o \leq \gamma \leq 10^o$. We obtain the reachable sets calculated using the LGG algorithm for each case. The unsafe region for the current scenario is the area containing redwood trees. For the current application the rectangle defined by the region $3600 \text{ ft} \leq x \leq 3800 \text{ ft}$ and $0 \text{ ft} \leq h \leq 360 \text{ ft}$ is deemed unsafe. We check whether the reachable sets for the small UAV reach the unsafe set or not.

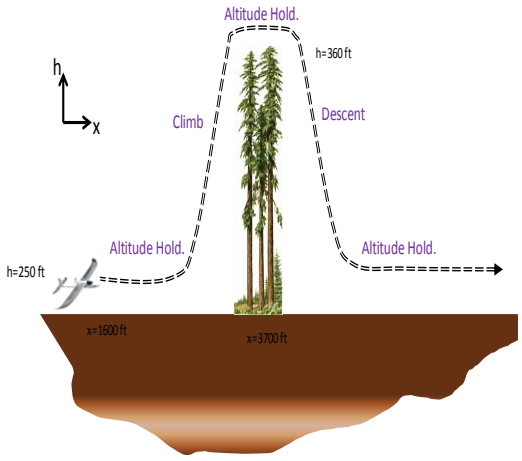


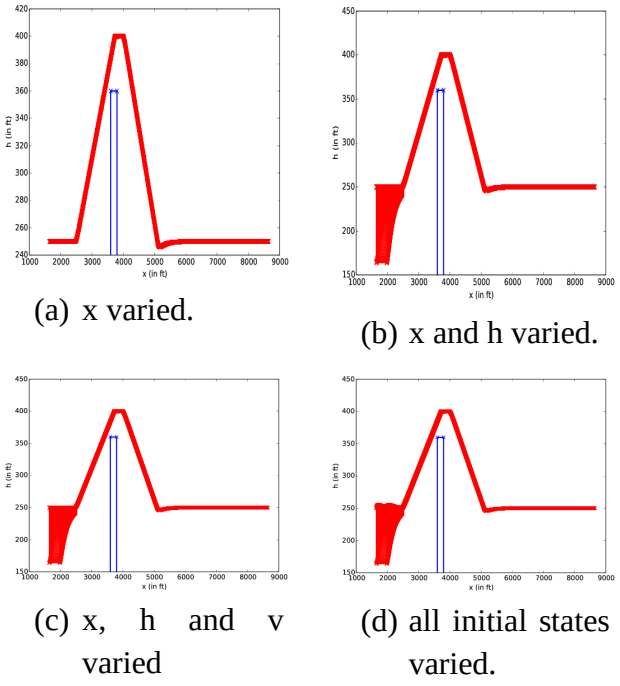
Fig. 9 Autopilot modes.

Fig. 10 shows the plots for the reachable sets. It can be seen that for all the cases the reachable sets avoid the unsafe region marked with blue boundary. Hence the autopilot logic is safe for the given UAV dynamics and the set of initial conditions defined above.

Further we investigate if the autopilot logic yields safe reachable sets in presence of uncertainty. Here, we add an uncertain parameter ν which is a uniform random process to the h-dynamics. Hence the h-dynamics is now represented as,

$$\dot{h} = V \cdot \gamma + \nu$$

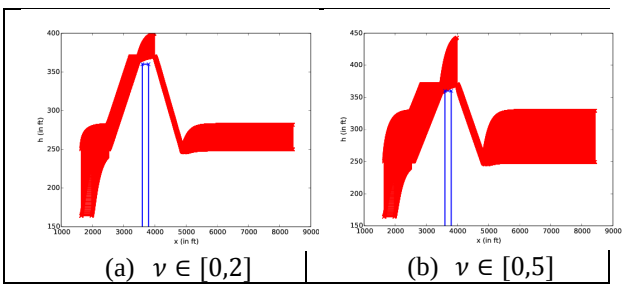
First it is assumed that $\nu \in [0,2]$. Further we vary all the initial states (x, h, V and γ) within the ranges as specified previously. We observe the reachable sets yielded and check if any violates the safety condition. It is observed that the autopilot still yields safe reachable sets and hence is safe. However, when we vary $\nu \in [0,5]$, it can be observed that the safety condition is breached by the reachable set. The reachable sets are depicted in Fig. 11. Note that the magnitude of external disturbance is an important factor in assuring the safety property, and careful estimate for the probable bounded uncertainty has a primary importance in precisely assessing the safety violation. The varying external disturbance cases shown in Fig. 11 conceptually confirms the importance of considering external disturbances in the framework of model checking and emphasizes the importance of research on obtaining a tight bound on the uncertainty.



(a) x varied. (b) x and h varied.

(c) x, h and v varied (d) all initial states varied.

Fig. 10 Reachable sets with the initial states varied.



(a) $\nu \in [0,2]$ (b) $\nu \in [0,5]$

Fig. 11 Reachable sets with $\boldsymbol{\nu}$ varied along with the initial states.

4 Extension to Onboard Model Checking

In general, model checking for hybrid systems in which discrete and continuous dynamics are coupled is a daunting task. However, if the system is fully autonomous, and the state transitions are continuous state dependent, then the hybrid system model can be approximated by an approximate quotient transition system (AQTS) that approximately partitions the state space of hybrid systems. In AQTSs, the continuous states are handled by computing invariants (or flow pipes) for each discrete state (mode). The transition between flow pipes is governed by state dependent guards. The AQTS creates a FSM representation of the original hybrid system so that standard model checkers can be used to verify its properties [8]. If the invariant partitions are represented using convex polyhedrons, then the approximation is called a polyhedral invariant hybrid automaton (PIHA) [5].

We choose to employ the PIHA model-based model checking procedure due to its speed and the popularity of the CheckMate tool [5]. Typically, hybrid model checking is performed on systems with individual modes in which UAV autopilots regulate the vehicle along the desired motion in the absence of external disturbances. If disturbances such as wind are not accounted for, however, the problem rapidly becomes trivial because the trajectory rapidly approaches the nominal trajectory. Ignoring the disturbances therefore dramatically underestimates the reachable set of the system, and model checking cannot identify errors that may exist in the actual flight control system.

In this research, we follow the approach in Ref. [5] and augment the reachable set of existing PIHA methods for linear systems with the H_∞ norm of the disturbance to the output transfer function. The H_∞ norm is often used in robust control to account for system disturbances and represents the steady-state bound of the worst case sinusoidal disturbance to the system. While this method is not strictly conservative, since transients and non-sinusoidal inputs could exceed the H_∞ norm, the H_∞ norm still gives an adequate bound for practical disturbances and can be used to identify previously undetected errors in hybrid systems.

The H_∞ norm is defined as follows:

$$\|\mathcal{G}(s)\|_\infty := \sup \frac{\|\mathcal{G}(s)d(s)\|}{\|d(s)\|} = \sup_\omega \bar{\sigma}\big(\mathcal{G}(j\omega)\big)$$

where $\mathcal{G}$ is the transfer function matrix of the system, d is the disturbance input to the system, and $\bar{\sigma}$ is the maximum singular value of $\mathcal{G}$. It can be computed rapidly and can be pre-computed for each mode of a linear hybrid system before computing the reachable set of the system, successfully accounting for the disturbance impact on the UAV behavior and bounded modeling errors. That is, the H_∞ norm augmentation used to account for environmental disturbances such as wind can also be used to address bounded UAV modeling errors.

The core idea here is to use the H_∞ norm to augment the computed reachable set for constructing the AQTS. For linear systems, the computation of the reachable set can also be done efficiently, so computation of the H_∞ norm augmented reachable set in real time for flight control systems should be feasible. In Fig. 12 the computed reachable set is shown for the initial set X_0. The H_∞ norm is used to augment the bound based upon the magnitude of the H_∞ norm and the disturbance for the current face of the reachable set. Note that, the H_∞ norm augmentation still strongly depends on the magnitude of disturbance since the augmented size is given by $\|\mathcal{G}(s)\|_\infty\|d(s)\|$. Therefore, it is indispensable to evaluate accurate disturbance bounds in order to assure the reachable set while

not making it too conservative. In particular, we have to take account of the practical statistics of wind disturbances at the UAV operating environment to assess the appropriate wind bounds. Fig. 13 depicts the reachability-based FSM for verifying the safety of UAV autopilots.

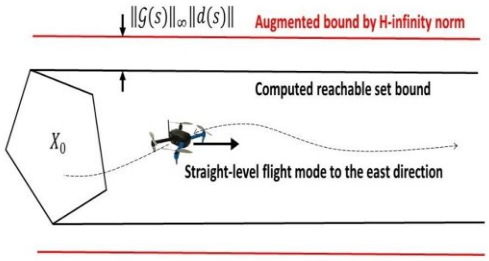


Fig. 12 Augmented reachable set

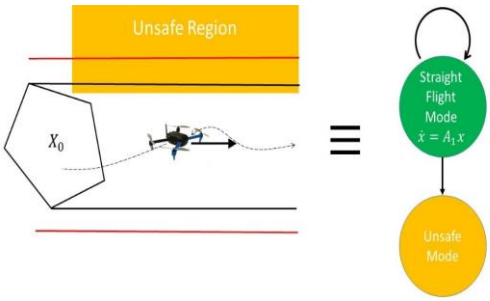


Fig. 13. Generation of an AQTS in case of unsafe UAV operation

One notable point of hybrid-system model checking in comparison with the traditional model checking in discrete states is that the state explosion problem, typical in software model checking, is a less concern as the safety is directly verified as the lack of overlap between the reachable set and an unsafe set. In hybrid model checking, the challenge is shifted towards how to compute the reachable set efficiently when the system dimension (of continuous dynamics) grows. Exact computation of reachability set computation for flight control systems has been generally deemed infeasible for onboard computation due to its numerical complexity [9]. A remedy for the computational burden is to employ an overapproximation so that the reachable set is overapproximated by simpler dynamics. This forms the basis for the online, onboard hybrid-system model checking algorithm.

Fig. 14 shows an example verification result with simple planar dynamics considered in Ref. [14]. The magenta line denotes a safety-violating trajectory in the presence of external disturbances. The details of the result can be found in Ref. [14].

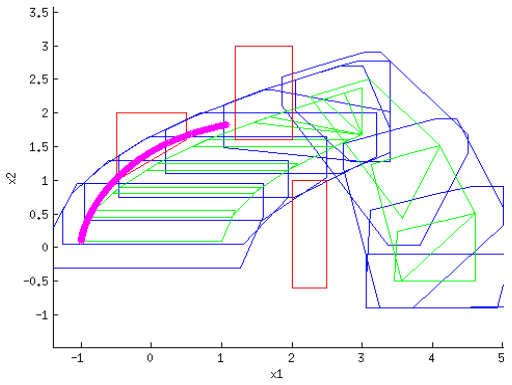


Fig. 14 Augmented reachable set (blue), unaugmented reachable set (green), and unsafe zones (red) for the 2-dimensional example system.

5 Conclusions and Future Work

This paper describes the research effort on applying hybrid-system model checking algorithms to the formal verification of small-scale UAV autopilots. The achieved result is a promising first step toward an efficient online model checker to be used onboard small-scale UAV. The research reveals that uncertainty arising in formal models and operation environment must be carefully incorporated into formal verification framework in order to avoid over-optimistic or over-pessimistic verification results.

The uncertainties inherent in real-time embedded system are expected to play a pivotal role in establishing the validity of online model checking methods for verifying the safety of flight control systems. Once it is established, it is expected that the approach will lead to non-iterative V &V techniques for emerging new

autonomous systems, which is an active research path pursued by the authors.

Acknowledgments

The research was supported under the NASA SBIR Phase I Contract No. NNX15CD22P. The authors thank the technical monitor Shaun McWherter at NASA Armstrong Flight Research Center for technical discussions.

References

[1] The Joint Planning and Development Office, "Unmanned Aircraft Systems (UAS) Comprehensive Plan: A Report on the Nation's UAS Path Forward", September, 2013.

[2] Waraich, Q. R.. Mazzuchi, T. A., Sarkani, S., and Rico, D. F., "Minimizing human factors in mishaps in unmanned aircraft systems," Ergonomics in Design:The Quarterly of Human Factors Applications, vol. 21, no.1, pp.25-32, 2013.

[3] Crum, V., Homan, D., and Bortner, R., "Certification challenges for autonomous flight control systems," In AIAA Guidance, Navigation, and Control Conference and Exhibit (pp. 16-19), August 2004.

[4] Wong, E., Schierman, J. D., Schlapkohl, T., and Chicatelli, A., "Towards Run-time Assurance of Advanced Propulsion Algorithms," 2014.

[5] Goppert, J., Gallagher, J. C., Hwang, I., and Matson, E., "Model Checking of a Flapping-Wing Micro-Air-Vehicle Trajectory Tracking Controller Subject to Disturbances," Robot Intelligence Technology and Applications 2, Springer-International Publishing, pp. 531-543, 2014.

[6] Jacklin, S. A., Lowry, M. R., Schumann, J. M., Gupta, P. P., Bosworth, J. T., Zavala, E., Kelly, J., Hayhurst, K., Belcastro, C. and Belcastro, C., Verification, validation, and certification challenges for adaptive flight-critical control system software, In *AIAA Guidance, Navigation, and Control Conference and Exhibit* (pp. 16-19), August 2004.

[7] Baier, C. and Katoen, J.-P., *Principles of model checking*, MIT press, Cambridge, 2008.

[8] Chutinan, A., and Krogh, B. H., Computational techniques for hybrid system verification, *IEEE Transactions on Automatic Control*, Vol. 48, No. 1, pp. 64-75, 2003.

[9] 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.

[10] The SpaceEx State Space Explorer, http://spaceex.imag.fr/ [accessed on 09/15/2015].

[11] Frehse, G., "PHAVer: Algorithmic verification of hybrid systems past HyTech," In *Hybrid Systems: Computation and Control*, pp. 258-273, Springer Berlin Heidelberg, 2005.

[12] Le Guernic, C., & Girard, A., "Reachability analysis of linear systems using support functions," Nonlinear Analysis: Hybrid Systems, Vol. 4, no. 2, pp. 250-262, 2010.

[13] Henzinger, T.A., "The theory of hybrid automata," In: *Proc. IEEE Symp. Logic in Computer Science*, LICS'96, New Brunswick, New Jersey, 27-30 July 1996, pp. 278–292, IEEE Computer Society,1996.

[14] Yang, B. J., Dutta, P., Tsai, J., Hwang, I., Yantek, S., Kwon, C., "Onboard model checking for small-scale UAV autopilots," Final report prepared under NASA SBIR Phase I Contract No. NNX15CD22P.

Contact Author Email Address

The first author can be contacted by mailto:jun.yang@optisyn.com

Copyright Statement