

Closing the Certification Gaps in Adaptive Flight Control Software

Stephen A. Jacklin¹

NASA Ames Research Center, Moffett Field, CA, 94035

Over the last five decades, extensive research has been performed to design and develop adaptive control systems for aerospace systems and other applications where the capability to change controller behavior at different flight conditions is highly desirable. Although adaptive flight control has been partially implemented through the use of gain-scheduled control, truly adaptive control systems using learning algorithms and on-line system identification methods have not seen commercial deployment. The reason is that the certification process for adaptive flight control software of this kind for use in national air space has not yet been decided. The purpose of this paper is to examine the gaps between the state-of-the-art methodologies used to certify conventional (i.e., non-adaptive) flight control system software and what will likely to be needed to satisfy FAA certification requirements. As these major certification gap areas are presented, a description of the current state of the verification methodologies, and what further research efforts will likely be needed to close the gaps remaining in current certification practices will be discussed. It is envisioned that closing the gaps will require advances in hybrid simulation methods, the development of new methods to analyze learning algorithm stability and convergence rates, the development of performance metrics for adaptive controllers, the application of formal software assurance methods, the application of on-line software monitoring tools for adaptive controller health assessment, and the creation of a certification case for adaptive system safety of flight.

Nomenclature

A, B	=	matrices
e	=	error vector
f	=	generic function
P, Q	=	generic symmetric positive-definite matrix
u	=	control vector
u_{Aug}	=	control augmentation vector
V	=	Lyapunov function
w	=	weight of neural network
x	=	state vector
y	=	measurement vector

I. Introduction

Over the last five decades, extensive research has been performed to design and develop adaptive control systems for aerospace systems and other applications where the capability to change controller behavior at different operating conditions is highly desirable. An adaptive controller changes its behavior by allowing the controller forward or feedback gains to be effectively adjusted once the controller has been placed into operation.¹⁻⁵ Because designing such a controller introduces many complexities, it is generally held to be good practice to use a non-adaptive or "classical" controller design if one can be found that delivers acceptable performance. This is because although proven techniques to evaluate the dynamic response and controller stability exist for non-adaptive controllers (e.g., root-locus, Bode plots, Nichols charts, etc.^{6,7}), techniques for adaptive systems are only yet in their infancy.

¹ Aerospace Engineer, Intelligent Systems Division, Moffett Field, CA 94035, Senior Member AIAA.

The only adaptive flight controllers that are certified for use in national air space are generally those that involve the use of a technique called gain scheduling.⁶ To implement this scheme, a non-adaptive controller is designed for a specific flight condition and vehicle configuration. The controller feed forward and/or feedback gains are then optimized for this flight condition. In simulation of the controller, the gains are tested at other flight conditions. If the performance is no longer good, new controller gains are computed. The perturbation of the flight conditions is repeated until the full flight regime is covered. The controller gains calculated for the various operating conditions and aircraft configurations are stored in a computer look-up table. The flight control computer is programmed to select the correct gains based on the current flight condition (airspeed, altitude, etc.) and vehicle configuration. The utility of this approach, from a certification point of view, is that each set of controller gains can be verified and validated by simulation and flight testing at the specific flight conditions for which they were chosen. Gain-scheduling thereby offers a means of partial adaptive control capability, while avoiding many of the problems associated with continuously changing controller gain values. By using a fine enough grid of flight conditions and vehicle configurations, virtually any pre-defined set of flight conditions can be handled using the gain scheduling method. The success of the gain scheduling method depends in large part on the degree to which the system can be characterized to operate at discrete flight conditions. For example, a few regimes could be airplane take-off, ascent phase, steady-state flight phase at a few altitudes, descent, and landing.

The focus of this paper does not concern the certification of gain-scheduled flight controllers, but rather a new breed of adaptive controllers that use system identification or some form of on-line learning to identify the optimal controller gain settings, system transfer matrices, or stability derivatives in real-time. Adaptive flight control systems of this type are currently being developed to help pilots recover from aerodynamic upset conditions,^{8,9} to regain vehicle handling qualities and stability in the event of aircraft damage or control surface failure¹⁰, to automatically fly vehicles autonomously in both air and space environments¹¹⁻¹⁵, to maintain vehicle performance during changing operating environments through use of neural networks,¹⁶⁻¹⁸ and to guide munitions to their targets¹⁹. These types of adaptive flight control systems must operate in highly non-linear and non-deterministic environments. Gain-scheduled control cannot be effectively used for these applications because the specification of all upset flight environments, the degree of control surface failure, or extent of aircraft or engine damage would require gain sets for an infinite number of flight conditions. Instead, for these applications, it is more efficient to assume the structure of the controller and use learning algorithms or on-line system identification methods to obtain the locally valid controller gain parameters.

Figure 1 provides a notional diagram of an adaptive control system to illustrate two possible ways to implement adaptive control.²⁰ In one way, a learning algorithm is used to compute flight control inputs to augment the controls produced by a non-adaptive flight controller. The other approach shown is to use a system identification algorithm to calculate gain parameters used by the flight controller. In this representation, the controller gains are not explicitly shown, but reside inside the flight controller box. The arrow through the box indicates the modification of the gain parameters inside the flight controller. It is possible to introduce adaptation using both methods at once and other schemes to introduce adaptive behavior are certainly possible.

Although the potential benefits of adaptive flight control systems are substantial, no adaptive flight control systems have been certified by the Federal Aviation Authority (FAA) for use in the national air space. The reason is that the means whereby adaptive flight control software can be routinely verified, validated, and certified for use in national air space has not yet been decided. As will be shown in the next section, the FAA has endorsed the use of RTCA DO-178 to provide certification guidelines for all flight software. Although software techniques exist to verify and validate conventional flight control software, the means to provide sufficient assurance of adaptive flight control software functionality, reliability, safety, and the absence of unintended functionality remains to be determined. If an adaptive flight controller is designed to make rapid and automatic adjustments to correct some diagnosed malfunction, it also has the ability to make a healthy aircraft un-flyable in the event of controller software failure. In a military application over restricted air space, such failures may be tolerated if not too high, but for a commercial application in civilian air space, such failures are unacceptable. Adaptive control systems with learning software will therefore never become part of the future unless it can be proven that the controller software is highly safe and reliable.

The objective of this paper is to examine the challenge areas that need to be addressed to enable the certification of adaptive flight control software for use in civilian air space. These are the gaps between the state-of-the-art methodologies used to certify conventional (i.e., non-adaptive) controller software and what is likely to be needed to satisfy FAA certification requirements.

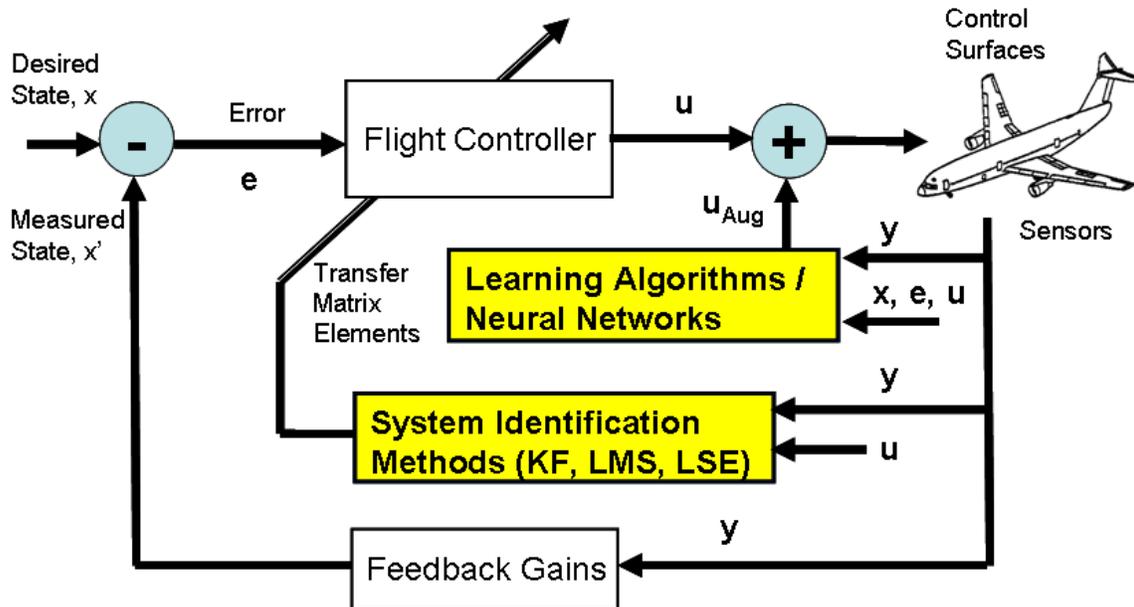


Fig. 1 Some ways to make a flight controller adaptive.

II. Where are the Certification Gaps ?

In the United States, the authority responsible for certifying flight control software is the Federal Aviation Administration (FAA). The FAA has stated in Advisory Circular 20-115B that all flight critical software must be developed according to the guidance provided in RTCA DO-178B²¹ or show compliance to airworthiness standards using some alternate means of compliance. Since the alternate means of compliance are undefined, the statements in DO-178B are generally regarded as certification requirements, rather than as mere guidelines. The objective of DO-178B is to help the aviation community develop flight software that can perform its intended functions while not negatively impacting other systems or the safety of aircraft operations. The document is maintained by the RTCA (Radio Technical Commission for Aeronautics) which is a private association of over 250 aeronautical organizations (established 1935).

DO-178B is not a process guide to software certification, but rather a description of what high-quality software development processes should be put in-place in order to create airborne software that performs its desired function. It levies no special requirements for adaptive flight control software; it is meant to apply to all airborne software. If it can be adequately demonstrated that these processes have been correctly and appropriately implemented, then any such flight software is, in principle, certifiable. This section highlights the specific guidelines recommended by RTCA DO-178B that, in the author's view, are potentially difficult for adaptive flight control software to fully satisfy. These areas indicate where the gaps reside and what infusion of new verification and validation techniques and methods might be needed to close them.

Table 1 provides a generalized summary of the basic DO-178B guidelines. This list has been generated by the author and is not intended to serve as a comprehensive index to the standard since it combines the intent of similar topics under general headings. However, it is useful for discussion purposes. The first column provides a list of the guideline categories or processes called for in DO-178B. The second column indicates which of these are more difficult for adaptive flight control software. So, for example, the first guideline, "provide an overview of the system and target application software", is marked "No" in the second column because this task is not more difficult to perform for adaptive flight control software than for conventional flight control software. As can be seen from the table, the majority of the guidelines do not present special problems for adaptive control software; they are equally difficult for non-adaptive flight software.

The rows marked with the "YES" designation in column 2 of Table 1 help identify the gap areas impeding the certification of adaptive systems. Defining software performance requirements is difficult due to lack of metrics to assess adaptive controller performance. Providing a software verification plan is difficult because it is hard to specify all possible failure modes of an adaptive controller in a highly nonlinear and non-deterministic environment,

and know the anticipated or expected behavior beforehand. Software requirements are difficult to specify because knowledge of how much learning is enough to provide good control is usually not known. All of these make determination of test cases to verify software function difficult.

Some airborne software developers argue that there is no guideline in RTCA DO-178B that cannot be satisfied using the same software assurance methods as are presently used for non-adaptive flight software.²² First, since DO-178B does not provide metrics to assess the adequacy of the verification and validation plans, it is somewhat of a judgment call to know when verification and validation plans are sufficient. This function is given to the DER (Designated Engineering Representative) who negotiates the certification process with the FAA and the software vendor. Second, it is argued that adaptive controllers behavior is only more difficult to prove because of the non-deterministic environments in which the software is designed to operate. This view, however, is not a systems approach; that is, the non-determinism cannot be ignored, regardless of the source.

The remaining sections of this paper discuss major gap areas in regards to the certification of adaptive flight control systems. Each of these sections describes a major gap area, presents a description of the present day state-of-the-art in these areas, and cites what further research efforts will likely be needed to close the gaps in the certification process. As shown in Table 1, most of these gaps generally fall under the DO-178B guidelines related to the development of the software verification plan. The software verification plan and verification tests provide a description of each activity in the software verification process. Generally, software verification is comprised of software review, software analysis, simulation, and testing. These activities may include the use of software programming checklists and formal software analysis and testing methods. Software analysis methods can include formal methods, static analysis, code reviews, traceability analyses, and coverage analyses.²⁰ The software verification plan establishes the rationale for the development of software test cases and methods.

Table 1: List of DO-178B Guidelines for Software Certification

DO-178B Guideline Topic	More Difficult for Adaptive Systems ?
Provide an overview of system and target application of software	No
Provide an overview of what the software does	No
Identify the software lifecycle	No
Define the software performance requirements	Yes
Provide a Software System Safety Assessment (SSA) Report. Lists all software failure modes and conditions and categorizes by failure severity	No (but more complex)
Provide a Software Development Plan	No
Provide a Software Verification Plan	Yes
Provide a Software Configuration Management Plan	No
Provide a Software Quality Assurance Plan	No
Define Software Requirement Standards	No
Define Software Design Standards	No
Define Software Code Standards	No
Define Software Requirements and all Derived Requirements	Yes
Software Design and Traceability Document	No
Provide Tool Qualification Data (e.g., autocoders, compilers)	No
Provide Source Code	No
Provide Executable Object Code	No
Provide Software Verification Test Cases and Procedures	Yes
Provide Software Verification Results	No
Provide Problem Reports	No
Provide Software Configuration Management Records	No
Provide Software Quality Assurance Records	No
Provide Plan for Software Aspects of Certification (PSAC)	Yes
Provide Software Accomplishment Summary	No
Provide Software Life Cycle Data	Yes

III. Gap in Defining Adaptive Controller Requirements

A critical gap which needs to be closed to facilitate certification is to develop procedures and methodologies to completely and correctly specify the design requirements of adaptive flight controllers. These software requirements define as precisely as possible what the software is supposed to do. DO-178B recommends that all requirements be written in a manner that allows them to be tested and may include such things as performance, precision, accuracy, and timing constraints. The requirements are frequently decomposed into derived requirements to address such considerations as computer speed, memory size, interfaces, and frequency of inputs and outputs. Non-adaptive controller performance requirements are usually specified by well-known metrics such as gain margin and phase margin.⁶ In contrast, the requirements for adaptive controller performance are usually only easy to express in overall desired properties, but not using precisely defined metrics.

A. Current State-of-the-Art

Most software life cycles (development through deployment) begin with an analysis to carefully define the software requirements as shown in Fig. 2. The left side shows the steps used to transform the requirements into software code. The right side shows the steps of software integration and testing to make sure the code ultimately satisfies the software requirements. The process of testing the performance of the final code against the defined software requirements is called software validation. This is the meaning of the second "V" in the often used acronym "V&V" for verification and validation. (The word validation is also often used in connection with model validation, but that is a very different process consisting of comparing the output of a model-based simulation against measured data.) Software verification is the analysis and testing processes used to ensure the software code does what it was designed to do. But proper verification testing does not imply successful validation testing.

The most common reason software fails validation testing is that the requirements are incompletely or poorly defined. Software developers may follow a verification process that proves that the software does exactly what it was designed to do algorithmically, but then discover that it does not meet the functional needs for control because of improper specification of the requirements at the outset. Such failures are very expensive to correct because when an error is found late in the validation process, the entire software design, verification, integration, and validation process shown in Fig. 2 must be performed all over again.

B. Further Research Needed

Certification of adaptive control systems will be significantly aided by the development of more precise ways to specify requirements and by the development of automated analysis tools to support verification and validation using model-based design simulation. The availability of modeling and simulation programs such as Matlab/Simulink²³ have encouraged the simulation of controller behavior prior to flight testing, and this has been the norm for many decades. Learning accuracy and controller stability are features usually tested in simulation, but rarely are the requirements themselves integrated into the testing.

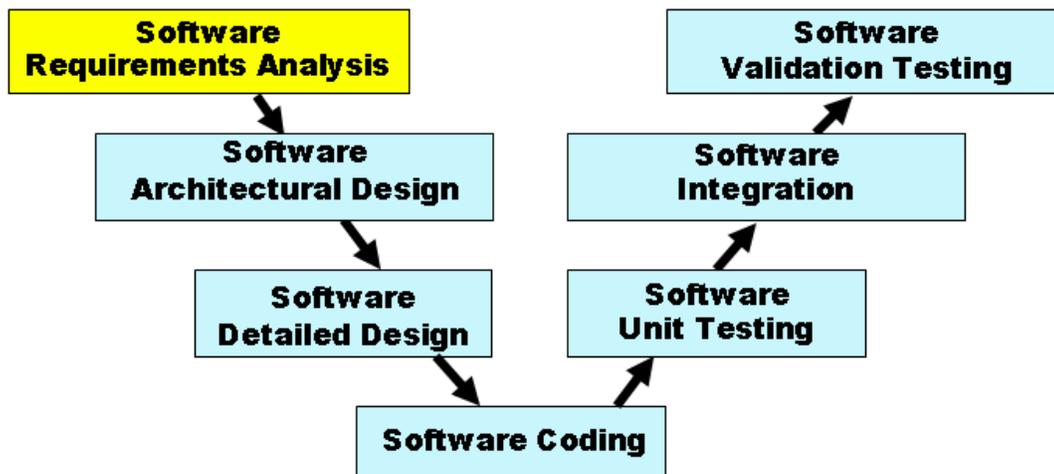


Figure 2. Software life cycle or development process.

One aspect that is emerging, however, is the concept of using aircraft, environment, and controller simulation models to fully evaluate software requirements at the outset of the controller design process.²⁴ The idea is to perform the end-stage system validation testing against the design requirements using the simulation model. The block diagram for this process is shown in Fig. 3. This figure shows an iterative software development path that features a model-based simulation. When the model is executed, the behavior of the controller can be observed and its performance assessed to see if it meets the software requirements. If it does not, the software architecture block diagrams can be modified and the tests repeated. The important feature of this approach is that it provides a means to perform end-stage validation of the requirements prior to writing any code for the actual target host computer. When complete, the software development for the actual control computer follows the same path as shown in Fig. 2, but now it is far more likely that costly end-stage validation problems will be encountered.

A gap that exists in this scheme is the lack of well-accepted performance metrics with which to both describe the requirements and create the necessary validation tests. For adaptive controllers, verification of learning algorithm or system identification adaptation function is of paramount importance. However, straight-forward tests of this type are difficult to establish. Reference 25, for example, holds that verifying the correct neural network weights have been learned for a closed-loop system represents an NP-complete hard (generally unsolvable) problem. Although the adaptive controller learning algorithm or system identification method can be coupled to the controller and the whole system tested as a unit in simulation, this ignores the obvious merit of being able to prove proper learning behavior as part of a certification process. One possibility may be to separately test learning algorithms and system identification methods in simulation using contrived aircraft models that have a unique, known solutions. The effects of measurement noise, process noise, persistent excitation, and ad hoc means to stop learning for low control errors might then be easily evaluated for their effect on learning. One such attempt at doing such experimentation for helicopter adaptive vibration and noise control algorithms is presented in Ref. 26. Such simulation cannot verify the learning behavior of over-parameterized systems, yet can provide proof that the basic learning algorithm was verified to work at least under ideal or known test conditions.

IV. Gap in Simulation Models for Adaptive Control

Since DO-178B presently allows certification credit to be obtained for both high-fidelity simulation testing as well as actual flight testing, it is highly likely that simulation will become an important part of the certification process for adaptive systems. As will be discussed below, some aspects of adaptive controller behavior can only be efficiently evaluated in simulation, rather than by complex mathematical analysis. A difficulty for certification, however, is the lack of uniform simulation methodologies for highly non-linear and non-deterministic flight control systems. To get simulation credit for an adaptive controller, some standardization will be required. Hence one gap area is held to be the definition of common simulation models to test various aspects of adaptive control system performance. Aside from certification, development of such models would also foster the creation of better controllers by providing a common benchmark for comparison.

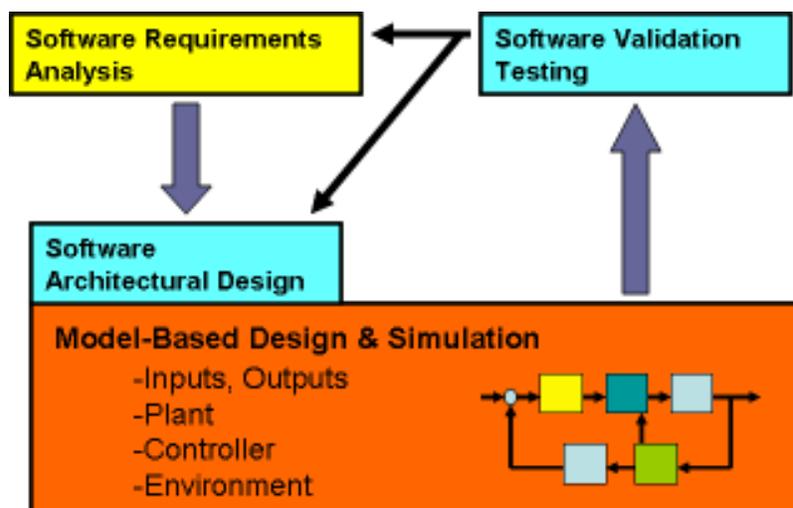


Figure 3. Model-based design methods tests candidate software designs for conformance to requirements prior to producing code for the target computer.

A. Current State-of-the-Art

Adaptive control system simulation can be done in a variety of ways. To keep simulation costs down, the first step in the simulation hierarchy is usually done on a desktop computer with a linearized model and linear aerodynamics representations. Desktop simulation is usually followed by more complex types of simulation. One of these is sub-scale or model testing using unmanned model aircraft. More commonly, the simulation complexity is first increased through the incorporation of non-linear aerodynamic and structural dynamic models. These simulations are usually run on a dedicated workstation computer platform. The next step level of fidelity includes using the actual target flight control computer in the simulation, as well as other hardware placed in the control loop such as cockpit control hardware and driving it with simulated sensor input. After this may follow motion-based simulation by a test pilot in a simulated cockpit environment that receives both visual and motion feedback. The last step is testing on the actual target flight vehicle.

The present state-of the art is to analyze adaptive system learning convergence and stability using simulation environments thought to provide enough fidelity to model significant nonlinear aerodynamics, dynamics, and other factors. Simulation provides a fairly rapid way to

- Evaluate and compare different learning algorithms,
- Tune control system gains and learning gains,
- Determine how much learning is actually being accomplished at each step of the simulation,
- Evaluate of the effect of process and measurement noises on learning convergence,
- Determine learning stability boundaries,
- Test algorithm execution speed on actual target flight computer,
- Conduct piloted evaluation of the learning system in a flight simulator, or,
- Simulate ad-hoc techniques of improving the learning process, such as adding persistent excitation to improve identification and convergence, or such as stopping the learning process after error is less than a specified amount.

A common practice done in simulation to evaluate the effect of controller gain selection is to use variations of the Monte Carlo analysis method.²⁷ In this method, the range of values for each parameter to be varied in simulation are determined beforehand. Within this range, a finite number of parameter test values are selected. If nothing is known about the parameter's expected value, then a uniform spacing throughout the parameter range is a logical choice. Alternatively, if the expected value of a parameter is known, then the test values can be more closely spaced near the expected value. Once all parameter test values have been selected, the matrix of simulation runs is comprised of every parameter varied in combination with all other parameter values. So, if there are three parameters that can take 5 unique values each, the number of simulation runs needed to evaluate the full matrix of possible combinations is 3^5 or 243. Even for non-adaptive controller simulation, the Monte Carlo method can be very time consuming considering the number of possible changes in the parameters needed to describe variations in the flight condition (airspeed, altitude, weight, etc.). When number of parameters becomes larger (as in the case of adaptive controllers), the number of simulation cases required can easily render the task of full Monte Carlo analysis intractable, except for very sparse parameter variations that leave large portions on the state space unexplored. From a certification standpoint, that is unacceptable.

B. Further Research Needed

A critical aspect of obtaining certification credit for simulation work will be proofs that the simulation fidelity is acceptably high so that important nonlinear effects are not missed. As has been cited above, the lack of common simulation model will likely inhibit both certification and the comparison of adaptive controller performance. The development of benchmark models is not an easy task and represents an important gap in the certification pathway for adaptive controllers.

Further work is needed to efficiently extend the important method of Monte Carlo analysis to adaptive controller evaluation. One method being studied by NASA under the Aviation Safety Program to help close the certification gap is to look at ways to extend the traditional Monte Carlo analysis method to assess the robustness of adaptive control systems. At the Langley Research Center, an analysis tool called RASCLE (for Robustness Analysis for Control Law Evaluation) has been developed to help explore combinations of learning system parameters and operating conditions.²⁸ The RASCLE simulation tool is used to interface with existing nonlinear simulations and incorporates search algorithms to uncover regions of instability with as few runs as possible. RASCLE uses a gradient algorithm to identify the direction in the uncertainty space along which the stability of the system is most

rapidly decreasing. RASCLE provides an intelligent simulation-based search capability that can be used in Monte Carlo simulation evaluations.²⁹ At the Ames Research Center, another approach to extend Monte Carlo analysis has been studied for the analysis of large, complex aerospace vehicles having highly coupled, nonlinear behavior and many degrees of freedom. In this approach, an algorithm has been developed to limit the number of combinatorial cases required of Monte Carlo analysis and to explore parameter interactions in a systematic (but not parametric) fashion. The data generated is automatically analyzed through a combination of unsupervised learning using a Bayesian multivariate clustering technique (AutoBayes) and supervised learning of the critical parameter ranges using the machine-learning tool TAR3, a treatment learner.³⁰ Covariance analysis with scatter plots and likelihood contours are used to visualize correlations between simulation parameters and simulation results.

The certification of adaptive controllers might also be aided by the development of probabilistic uncertainty models for simulation in order to quantify their robustness. Reference 31 discusses the development of probabilistic uncertainty models to assess the effect of parameter variations on controller stability. By making various cross-plots using the controller parameters, the plot space can be divided into regions called the Failure Domain and the Admissible Domain. By varying the parameters, probabilistic uncertainty methods can define a set of plants and associate a weight (or probability) for each. This then facilitates a search for a robust controller by being able to quantify how far away the controller class is from instability or some other problem by the parameter variation method known as homothetic deformations.

The certification of adaptive systems will also likely be aided by the development of hybrid simulation models that can model the full control system as an integrated whole, rather than each system in isolation from each other. Hybrid models can be very complex and there are different definitions of “hybrid model” in usage today. Reference 32 defines a hybrid adaptive controller as a combination of direct and indirect adaptive control. Hybrid systems have also been used to define systems comprised of finite state executive controllers coupled to a continuous domain adaptive controllers.^{20,33} A related characterization of a hybrid system is one in which a finite state controller using a continuous learning algorithm is coupled to a discrete model of the (normally) continuous environment.³⁴ This characterization might seem odd, but using a hybrid model to represent the continuous state vector as a discrete variable is being studied as a means of leveraging the power of model checking software such as SPIN³⁵, NuSMV³⁶, and JPF2^{37,38}. Model checking is a technique by which a finite state system model can be exhaustively explored to make sure the system never reaches an unacceptable state. The method relies on being able to express adaptive controller safety properties as assertions in temporal logic and having a suitable approximation function to convert the continuous variables into discrete values.³⁴

V. Gap in Proving Learning Stability and Convergence

The critical gap in the verification plans for adaptive control systems is the lack of procedures that can reliably verify that the learning algorithm or system identification method learns correctly and converges to the correct solution in an acceptable time. The provision of this guarantee is probably the most important aspect of a viable verification plan for an adaptive control system. Other than learning and system identification, there is no difference between adaptive and non-adaptive controllers, so proving the learning process is stable and convergent at all flight conditions and vehicle configurations is of paramount importance.

Consider the neural network neuron shown in Fig. 4. A learning algorithm of the type

$$\begin{aligned}\Delta w_i &= f(e) \\ w_i(k+1) &= w_i(k) + \Delta w_i\end{aligned}$$

is typically used to update the weight values based on some error metric, e . This error could be formed as the difference between the measured state of the aircraft and the state predicted using the model parameters, w . The function f depends on the learning algorithm used (e.g., steepest descent, Gauss-Newton, Levenberg-Marquardt, etc.). It can be seen in Fig. 4 that if the six weighted inputs are summed together to form one output, the values of the weights offering a correct solution is not necessarily unique, nor even guaranteed to exist. System identification methods for transfer matrix identification are alike in this respect because the neuron shown in Fig. 4 is mathematically equivalent to a row-column matrix multiplication operation. There are many excellent texts³⁹ that have analyzed the necessary conditions for a unique solution. Since there are six inputs and one output, the minimum requirement is that the outputs be known for at least 6 linearly independent input vectors. Then six equations in six unknowns exist, and values of the weights can be determined, subject to the effects of noise on the

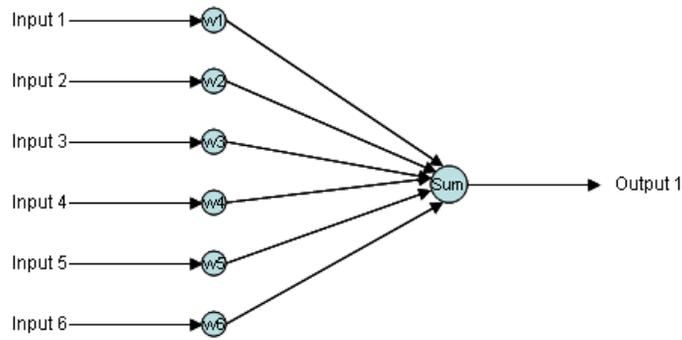


Figure 4. Neural network neuron.

measurements. However, if the second input is the square of the first input, and the third input is the product of the fourth input and the first input, and so on, the inputs are not linearly independent and the existence of a solution may be possible only in the sense of finding a set of weights that is optimal according to some specified criteria. Neural networks often employ such input combinations to offer superior curve fits to nonlinear data.

A. Current State-of-the-Art

Mathematical proofs of adaptive controller stability generally seek to show that the vehicle state returns to a neighborhood about the undisturbed state for every defined disturbance. The most commonly used proof of this is based on Lyapunov's second method.^{4,40-42} For linear time invariant systems of the form,

$$\dot{x} = Ax$$

where x is the state vector and A is a matrix, the Lyapunov method states that the system is stable (will return to the origin) if a Lyapunov function $V(x)$ can be found that is always positive and that has a time derivative that is always negative or equal to zero, or,

$$V(x) > 0$$

$$\dot{V}(x) \leq 0$$

Unless much is known about the dynamics of the combined learning system and adaptive controller, the Lyapunov function is usually chosen as the simple quadratic function,

$$V(x) = x^T Px$$

and the system is said to be stable if, and only if, given any symmetric positive-definite matrix Q , there exists a symmetric positive-definite matrix P , which is the unique solution of the set of

$$A^T P + PA = -Q$$

equations.⁴ Although finding a Lyapunov function has in the past been somewhat of a cumbersome trial and error process, recent advances in semidefinite programming and semialgebraic geometry have afforded more of an algorithmic procedure to find valid Lyapunov functions through the use of the sum of squares (SOS) method.⁴³

B. Further Research Needed

From a certification perspective, a weakness of the Lyapunov approach to prove stability is that it requires a polynomial representation of the plant (A matrix) for all flight conditions of interest. If the values of this representation change, perhaps due to aircraft damage, then nothing can be said about controller stability based on

knowledge of the previous plant matrix. More importantly, the Lyapunov analysis only guarantees the ultimate stability of the learning algorithm; the proof does not guarantee how fast the system returns to the origin. In adaptive controller parlance, this means that Lyapunov proofs cannot guarantee the rate of learning convergence. This is an important point for system performance, because if learning happens too slowly, an adaptive controller may be rendered ineffective for the control task at hand.

Although the Lyapunov approach to stability determination rests on a firm mathematical foundation, a problem using this method for the certification of adaptive control systems is that there is a gap between the mathematician's knowledge and the understanding of the certification official who needs to see a more easily understandable explanation that offers easy assessment. A mathematical proof that is understood only by experts in the control field will fail to inspire certification authorities that all due diligence has been done to ensure safety. Moreover, since these proofs depend on knowledge of the "A" and "B" matrices of the control system representation,

$\dot{x} = Ax + Bu$, and these may actually change with time (e.g., with aircraft damage), Lyapunov proofs may well lack the sufficient conditions necessary to ensure adaptive controller stability. A similar criticism holds for the method of attempting to prove regions of controller stability through the use of barrier certificates.^{44,45} Like the Lyapunov approach, this method seeks to prove that the state trajectories starting from a given set of initial conditions never reach an unsafe region. It is also fairly difficult to understand without extensive mathematical training. The barrier certificate represents a guaranteed upper bound on the probability that the system trajectories do not reach the unsafe set of states. However, application of the method requires that the plant be expressed in polynomial form and that the boundaries of the unsafe states also be expressed in the same way.

To help close this gap in the certification process, it is held that Lyapunov stability analysis needs to be augmented by the development of metrics to measure adaptive control system stability and robustness. Metrics that can assess how far away the system is from instability are needed that are analogous to the gain margin and phase margin metrics used to assess non-adaptive, linear controllers. The metrics need to be ones that can be easily assessed and related to physical quantities that can be easily measured. A certification test to assess "how far away" from the true values the learned or identified parameters of the control system are would be particularly valuable.

The lack of methods to evaluate the effects of measurement and process noise on learning convergence rates is another gap. In theory, it is usually possible to show that learning convergence can be related to the degree of linear independence present in the control input vectors.²⁶ In practice, measurement noise and/or process noise removes this independence as the controller converges to a near optimal control solution. In this case, the learning algorithm or system identification method will diverge because the learning process seeks to find a relationship between small (real) changes in the control vector to changes in the measurements resulting largely from measurement noise. Since this relationship is a random variable (or like the noise distribution), the dynamics of the plant become lost. The effect of noise on learning and identification performance as well as ad hoc approaches such as disabling learning during period of low control error are difficult to analytically evaluate, and so simulation tools offer hope.

Yet another gap is that methods to find acceptable gains for stable learning must be found other than the time intensive trial and error process. It has been found through analysis that high adaptation gains will often lead to high frequency oscillation in the tracking errors, especially in poor signal to noise environments.^{4,32} This means that acceptable gains for stable learning performance must be found by trial and error, and some means of high-fidelity simulation to choose the compromise between rapid learning and oscillatory tracking would be useful.³²

It is also known that convergent adaptive control system learning or system identification convergence usually requires persistent excitation.² This problem generally occurs when the controller computes the correct optimal control before the system identification or learning method completely converges. If the controller is able to find a control vector that effectively nulls the error between the desired state and measured state, then updates to the learned or identified values based on that error signal will also tend to zero and the system will not learn until the control error becomes higher. A persistent excitation signal added to the control signal yields more convergent learning, but at the expense of poorer steady-state controller performance. An alternative is to disable the learning process when the control error becomes low, but this requires further study. References 32 and 26 provide detailed insight to this problem. Better simulation methods need to be developed to enable control engineers select the correct degree of persistent excitation and to acquire a knowledge base from which to judge the how this excitation effects the behavior of the actual control system.

VI. Gap in On-Line Monitoring Tools

The development of on-line tools to monitor adaptive controller performance after deployment may form a necessary part of a certification plan. These tools would offer a way to make a system health management assessment to indicate when controller learning is acceptably good or when the aircraft is beyond control. On-line

tools of this nature may be required in the event analytical or simulation testing is unable to verify convergent, stable learning from every possible flight condition and vehicle state..

A. Current State-of-the-Art

At the present time, tools to assess the performance of adaptive flight control systems are an area of great research interest. One example of an on-line software assurance tool is the Confidence Tool⁴⁶ developed under the NASA Aviation Safety Program. This tool provides a useful metric to assess neural network weight convergence using a Bayesian approach. The Confidence tool is a dynamic monitor which checks the output values of a neural network based controller and determines a confidence measure to evaluate the correctness of the neural network weights based on a statistical model of the learning system. The Confidence tool dynamically calculates a metric of neural network learning called the confidence measure. This value can be used as a metric related to control system health.

An on-line monitoring tool for in-flight stability evaluation has been developed by the NASA Dryden Flight Research center for the X-38 crew return vehicle.⁴⁷ This method introduced a small-amplitude tailored-force excitation to the elevator and rudder control surfaces at specific frequencies. The banded frequency response was then used to calculate the stability margins of the flight control system using a modification of the method in Ref. 48. A recursive Fourier transformation was used to make the method compatible with real-time calculation. The stability calculated by the on-line method was shown to compare well to results obtained using the X-38 nonlinear simulation. For the certification of adaptive control systems, such a method might be extended to evaluate adaptive control system stability and make on-line predictions of unstable regions in order to avoid them.

B. Further Research Needed

While research is needed to develop on-line, adaptive controller health assessment tool, another need is to extend formal methods to analyze executive flight controller software of the real-time operating system. Formal methods have almost exclusively been developed for the assessment of finite state systems, rather than for the continuous domain of adaptive control. Nevertheless, formal analysis tools may be extendable to the verification of executive, outer-loop (or supervisory) adaptive controllers. For example, Ref. 49 describes an application of the NASA Ames Java PathFinder model checker to the control the guidance of a robotic vehicle. Using compositional verification to verify that the interface logic between components will function properly when integrated together is a valuable concept that works well when combined with model checking of large systems. The difficult part of extending these methods to adaptive control is finding ways to represent the continuous state vector and system safety assertions as finite state temporal logic assertions. If a hybrid model abstraction for the adaptive system can be found, then the adaptive control states become finite values. This allows for the recognition of previous “states” in the model checking sense of the word, and hence an exploration of the continuous model checking space becomes possible. Of course, this search is exhaustive only to the extent the approximation function is valid. If the approximation function is too coarse, important states will likely be missed.

VII. Gap in Adaptive Controller Certification Plans

The software development process described in RTCA DO-178B for airborne software recommends that the verification and validation plans are developed before any code is written. Verification and validation plans for certifiable software need to provide a test matrix together with an explanation why each test point has been chosen and how together all of the test points will provide adequate test coverage. DO-178B recommends that the report should include a description of the conditions under which each test is to be performed and state the pass/fail criteria. Step by step instructions for performing each test are to be provided along with instructions with how to evaluate the test results. DO-178B stresses that it is important that these procedures and criteria be developed prior to the actual testing.

A difficulty with even non-adaptive controller development is that the usual path of controller software development is analysis immediately followed by desktop simulation and then other higher-fidelity simulation test, possibly including sub-scale testing. The simulation tests are not formal software verification tests, but rather just attempts to evaluate the performance of the learning algorithm and controller performance. The dissonance between this ad hoc development process and the orderly process advised by DO-178B indicates the need for a better certification plan for adaptive control systems development that is more attuned to real-world development practices.

A. Current State-of-the-Art

It is not possible to mention all of the on-going efforts by industry and government offices to develop adaptive flight control systems.^{10,13,15,16,35,50-53} The Air Force VVIACS (Verification and Validation of Intelligent and Adaptive Control Systems)⁵⁴ and the NASA IRAC (Intelligent Resilient Adaptive Control)⁵⁵ represent multi-year programs with industry partners that have been initiated to define methodologies and test procedures for adaptive flight control systems. Although the VVIACS program has ended, the IRAC Program continues to be sponsored by the NASA Office of Aviation Safety. The goal of the IRAC Program is to conduct research to advance the state of aircraft flight control to provide onboard control resilience for ensuring safe flight in the presence of unforeseen, adverse conditions. The objective is to advance the state-of-the-art of adaptive controls as a design option to provide enhanced stability and maneuverability margins for safe landing. It is anticipated that the outcome of the IRAC project research will be a set of validated, multidisciplinary integrated aircraft control design tools and techniques for enabling safe flight in the presence of adverse conditions such as structural damage, control surface failures, or aerodynamic upsets. With regard to the certification of adaptive flight control systems, it is hoped that the analysis, simulation, sub-scale and full-scale flight tests of this research program will help form the basis for a valid Plan for Software Aspects of Certification (PSAC) for adaptive flight control systems, or perhaps even the basis for a better certification process for adaptive flight controllers.

B. Further Research Needed

Further research is needed to determine the best practices for a workable adaptive control system certification plan. Besides the gaps identified above, there are several other aspects which need to be considered as well. A very practical aspect of a flight control systems is that DO-178B advises that safety-critical software should provide a measure of software redundancy and fault tolerance. The preferable level of redundancy is two systems doing the same thing, but using different calculation methods to arrive at the same answer. This is referred to in DO-178B as redundancy achieved by using dissimilar implementations. A problem with using this technique with adaptive flight controller software is that the dissimilar implementations could take different control trajectories to achieve the same end state and yet not be comparable along the way. Another aspect of a viable certification plan is to address the level of fault tolerance required of the control system. Verification of software health management software operating on partitioned real-time operating systems (RTOS) to ensure any failures in the controller remain isolated, will be critically important for both aeronautics and space applications.⁵⁶ Guidelines for these systems in not provided in DO-178B, but rather in ARINC-653.⁵⁷

More research needs to be conducted in cooperation with the FAA to explore alternative means of compliance to DO-178B. Once a sufficient set of best practices for the verification and validation of adaptive flight control systems becomes available, it is possible that these practices could be grouped to form a Safety Case argument for adaptive flight control systems. Safety cases have been created for certification of nuclear industry in Europe and off-shore oil refineries in Australia.^{58,59} A safety case is a document that identifies all hazards and risks, describes how the risks are controlled, and describes the safety management plan to ensure the controls and guidelines are effectively and consistently applied. The safety case represents a collection of processes to ensure all identified risks are mitigated. In this way, the development of stability analysis methods for adaptive controllers, metrics for adaptive controller performance (or learning), hybrid high-fidelity simulation methods, the usage of formal methods, and other technologies mentioned above might become part of the safety case. In essence, the safety case argues for software certification on the basis that every best practice to ensure safety has been followed. Whether or not this is the same thing as proving the system is safe is a valid question. In fairness, however, any certification procedure fulfilling the spirit of the DO-178B guidelines might also end up not being safe.

VIII. Summary

This paper has provided an examination of the gaps between current state-of-the-art methodologies used to certify airborne software and what is likely to be needed to satisfy FAA airworthiness requirements for the certification of adaptive flight control systems. Adaptive controllers (as defined herein) use system identification or some form of on-line learning algorithm to identify optimal controller gain settings, system transfer matrices, and/or stability control derivative matrices in real-time. The gaps presented in this paper include the lack of a certification plan or process guide, the need to develop verification and validation tools and methodologies to analyze adaptive controller stability and convergence, the need to develop metrics to evaluate adaptive controller performance at normal and off-nominal flight conditions. This paper has also presented for each gap area a description of the present day state-of-the-art and what further research efforts will likely be needed to close the gaps remaining in current certification practices. The areas addressed include advances in hybrid simulation methods, the development

of new methods to analyze learning algorithm stability and convergence rates, the development of performance metrics for adaptive controllers, the application of formal software assurance methods, the development of on-line software monitoring tools for adaptive controller health assessment, and the creation of a certification case for adaptive system safety of flight.

References

- ¹Åström, K. J. and Wittenmark, B. Adaptive Control, 2nd ed. Reading, MA: Addison-Wesley, 1995.
- ²Ioannou, P. A. and Sun, J. Robust Adaptive Control. Englewood Cliffs, NJ: Prentice Hall, 1996.
- ³Landau, I. D., Adaptive Control: The Model Reference Approach, New York, NY: Marcel Dekker, 1979.
- ⁴Narendra, K. S. and Annaswamy, A. M., Stable Adaptive Systems, Englewood Cliffs, NJ: Prentice Hall, 1989.
- ⁵Sastry, S. and Bodson, M., Adaptive Control: Stability, Convergence, and Robustness, Englewood Cliffs, New Jersey: Prentice-Hall, 1994.
- ⁶Ogata, K., Modern Control Engineering, 4th Edition, Pearson Education, Nov 2001.
- ⁷Bryson, A. E. and Ho, Y. C., Applied Optimal Control, Taylor and Francis, 1975.
- ⁸Kaneshige, John, and Gundy-Burlet Karen, "Integrated Neural Flight and Propulsion Control System," Proceedings of the AIAA Guidance, Navigation, and Control Conference, AIAA-2001-4386, August 2001.
- ⁹Williams-Hayes, P.S., "Flight Test Implementation of a Second Generation Intelligent Flight Control System," Technical Report NASA/TM-2005-213669, 2005.
- ¹⁰Nguyen, N., Krishnakumar, K., Kaneshige, J., and Nespeca, P., "Dynamic and Adaptive Control for Stability Recovery of Damaged Asymmetric Aircraft," Proceedings of AIAA Guidance, Navigation and Control Conf., Keystone, CO, Aug. 2006. AIAA 2006-6049.
- ¹¹Kaneshige, J., Bull, J., and Totah, J., "Generic Neural Flight Control and Autopilot System," AIAA-2000-4281.
- ¹²Hall, R., Barrington, R., Kirchwey, K. and Alaniz, A., "Shuttle Stability and Control During the Orbiter Repair Maneuver," AIAA Guidance, Navigation and Control Conference and Exhibit, 2005, AIAA 2005-5852.
- ¹³Johnson, E. N. and Calise, A. J., "Limited Authority Adaptive Flight Control for Reusable Launch Vehicles," Journal of Guidance Control and Dynamics, vol. 26, pp. 906-913, 2003.
- ¹⁴Hovakimyan, N., Kim, N., Calise, A.J., Prasad, J.V.R., and Corban, E.J., "Adaptive Output Feedback for High-Bandwidth Control of an Unmanned Helicopter," AIAA Guidance, Navigation and Control Conference, AIAA-2001-4181, 2001.
- ¹⁵Lavretsky, E. and Wise, K., "Adaptive Flight Control for Manned/Unmanned Military Aircraft," Proceedings of American Control Conference, Portland, OR, June 2005.
- ¹⁶Rysdyk, R.T. and Calise, A.J., "Fault Tolerant Flight Control via Adaptive Neural Network Augmentation," AIAA Guidance, Navigation, and Control Conference, AIAA-1998-4483, 1998.
- ¹⁷Johnson, E.N., Calise, A.J., El-Shirbiny, H.A., and Rysdyk, R.T., "Feedback Linearization with Neural Network Augmentation Applied to X-33 Attitude Control," AIAA Guidance, Navigation, and Control Conference, AIAA-2000-4157, 2000.
- ¹⁸Calise, A. J., Lee, S., and Sharma, M., "Development of a Reconfigurable Flight Control Law for the x-36 Tailless Fighter Aircraft," Proceedings of the AIAA Guidance, Navigation, and Control Conference, Denver, CO, Aug. 2000.
- ¹⁹Calise, A. J., Sharma, M., and Corban, J. E. "Adaptive Autopilot Design for Guided Munitions," Journal of Guidance, Control, and Dynamics, vol. 23, pp. 837-843, 2000.
- ²⁰Jacklin, S. A., Schumann, J., Gupta, P., Richard, M., Guenther, K., and Soares, F., "Development of Advanced Verification and Validation Procedures and Tools for the Certification of Learning Systems in Aerospace Applications," Proceedings of the AIAA Infotech@Aerospace Conference, Crystal City, VA, Sept. 2005.
- ²¹Software Considerations in Airborne Systems and Equipment Certification, Document No RTCA (Requirements and Technical Concepts for Aviation) /DO-178B, December 1, 1992.
- ²²Santhanam, V. "Can Adaptive Flight Control Software be Certified to DO-178B Level A?", NASA and FAA Software and CEH Conference, Norfolk, VA, July 26-28, 2005.
- ²³MATLAB, The MathWorks Inc. <http://www.mathworks.com/products>.
- ²⁴Pires, C., Cannon, H., Christa, S., Limes, G., Dorais, G., Branson, M., Kulkarni, N., Viazzo, D., Brown, J., Kelly, A., and Logan, M., "Verification and Validation Strategies and Vision," NASA Ames Live Webinar for The Mathworks, June 26, 2008.
- ²⁵Garey, M. R., and Johnson, D. S., Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman and Company, 1979.
- ²⁶Jacklin, S. A., "Comparison of Five System Identification Algorithms for Rotorcraft Higher Harmonic Control," NASA TP 1998-207687, May 1998.
- ²⁷Belcastro, Christine, and Belcastro, Celeste, "On the Validation of Safety Critical Aircraft Systems, Part I: Analytical & Simulation Methods," Proceedings of AIAA Guidance Navigation and Control Conference, Austin TX, August 2003.
- ²⁸Bird, R.: RASCLE Version 2.0: Design Specification, Programmer's Guide, and User's Guide. Baron Associates, Inc., February, 2002.
- ²⁹Belcastro, Christine, and Belcastro, Celeste, "On the Validation of Safety Critical Aircraft Systems, Part I: Analytical & Simulation Methods," Proceedings of AIAA Guidance Navigation and Control Conference, Austin TX, August 2003.
- ³⁰Schumann, J., Burlet, K. G., Pasareanu, C., Menziers, T., and Barrett, T., "Tool Support for Parametric Analysis of Large Software Simulation Systems," submitted to Automated Software Engineering Conference, L'Aquila, Italy, Sept. 2008.

- ³¹Crespo, L. G. and Kenny, S. P., "Robust Control Design for Systems with Probabilistic Uncertainty," NASA/TP-2005-213531, March 2005.
- ³²Nguyen, N., and Jacklin, S. A., "Neural Net Adaptive Flight Control Stability, Verification and Validation Challenges, and Future Research," IJCNN Conference, Orland Florida, 2007.
- ³³Tomlin, C. and Greenstreet, M. R., editors. Hybrid Systems: Computation and Control, 5th International Workshop, HSCC 2002, Proceedings, volume 2289 of Lecture Notes in Computer Science. Springer, 2002.
- ³⁴Clarke, E. M., Fehnker, A., Han, Z., Krogh, B. H., Stursberg, O., and Theobald, M., "Verification of Hybrid Systems based on Counterexample-Guided Abstraction Refinement," in Tools and Algorithms for the Construction and Analysis of Systems, 9th Intl. Conf., TACAS 2003, pages 192-207. Springer, 2003.
- ³⁵Holzmann, G. J., The Spin Model Checker Primer and Reference Manual, Addison-Wesley, Boston, MA, 2004.
- ³⁶McMillan, K., Symbolic Model Checking. Kluwer Academic Publishers, Boston, MA, 2003.
- ³⁷Visser, W., Havelund, K., Brat, G., Park, S., and Lerda, F., Model Checking Programs, Kluwer Academic Publisher, 2002.
- ³⁸Havelund, K., "Using Runtime Analysis to Guide Model Checking of Java Programs," SPIN Model Checking and Software Verification, Vol. 1885 of Lecture Notes in Computer Science. pp. 245-264, Springer, 2000.
- ³⁹Kailath, T., Linear Systems, Prentice-Hall Information and System Sciences Series, 1979.
- ⁴⁰Wise, K. A., Lavretsky, E., and Hovakimyan, N., Robust and Adaptive Control Workshop, American Control Conference, Seattle, WA, June 11-13, 2008.
- ⁴¹Khalil, H. K., Nonlinear Systems, Prentice Hall, 3rd Edition, 2001.
- ⁴²Slotine, J.-J. E. and Li, W., Applied Nonlinear Control, Prentice Hall, New Jersey, 1991.
- ⁴³Prajna, S., Papachristodoulou, A., and Parrilo, P. A., "Introducing SOSTOOLS: A general purpose sum of squares programming solver," in Proceedings IEEE Conference on Decision and Control, 2002, available at <http://www.cds.caltech.edu/sostools>.
- ⁴⁴Prajna, S., Jadbabaie, A., and Pappas, G. J., "Stochastic Safety Verification Using Barrier Certificates," Proceedings of 43rd IEEE Conference on Decision and Control, Atlantis, Paradise Island, Bahamas, December 2004.
- ⁴⁵Prajna, S., and Jadbabaie, A., "Safety Verification of Hybrid Systems Using Barrier Certificates," in Hybrid Systems: Computation and Control. Heidelberg: Springer-Verlag, 2004.
- ⁴⁶Jacklin, S. A., Schumann, J., Bosworth, J., Williams, P., and Larson, D., "Test Results of a Tool and Method for In-Flight, Adaptive Control System Verification on a NASA F-15 Flight Research Aircraft," 7th World Congress on Computational Mechanics, Los Angeles, CA, July 2006.
- ⁴⁷Bosworth, J. T., and Stachowiak, S. J., "Real-Time Stability Margin Measurements for X-38 Robustness Analysis," NASA/TP-2005-212856, Feb 2005.
- ⁴⁸Bosworth, J. T. and Burken, J. J., "Tailored Excitation for Multivariable Stability-Margin Measurement Applied to the X-31A Nonlinear Simulation," NASA TM-113085, 1997.
- ⁴⁹Scherer, S., Lerda, F., and Clarke, E., "Model Checking of Robotic Control Systems," Proceedings of ISAIRAS 2005 Conference, Munich, Germany, Sept. 5-8, 2005
- ⁵⁰Cao, C. and Hovakimyan, N., "Design and Analysis of a Novel L1 Adaptive Control Architecture, Part I: Control Signal and Asymptotic Stability," Proceedings of American Control Conference, pages 3397-3402, Minneapolis, MN, June 2006.
- ⁵¹Krishnakumar, K., Limes, G., Gundy-Burlet, K., and Bryant, D., "An Adaptive Critic Approach to Reference Model Adaptation," AIAA Guidance, Navigation, and Control Conference, AIAA-2003-5790, 2003.
- ⁵²Tao, G., Chen, S. H., Fei, J. T., and Joshi, S. M., "An Adaptive Actuator Failure Compensation Scheme for Controlling a Morphing Aircraft Model," Proceedings of the 42nd IEEE Conference on Decision and Control, Maui, Hawaii, 2003.
- ⁵³Liu, Y., Tang, X. D., Tao, G., and Joshi, S. M., "Adaptive Failure Compensation for Aircraft Tracking Control Using Engine Differential Model," Proceedings of the 2006 American Control Conference, Minneapolis, MN, June 2006.
- ⁵⁴Buffington, J. M., Crum, V., Krogh, B., Plaisted, C., and Prasanth, R., "Verification and Validation of Intelligent and Adaptive Control Systems," 2nd AIAA Unmanned Unlimited Systems Conference, San Diego, CA, Sept. 2003.
- ⁵⁵Totah, J., Krishnakumar, K., and Viken, S., "Stability, Maneuverability, and Safe Landing in the Presence of Adverse Conditions," Report of NASA Integrated Resilient Aircraft Control Project, April 13 2007.
- ⁵⁶Jacklin, S. A., Lowry, M. R., Hayden, S., Day, L., Hicks, K. A., and Brinza, D. E., "Vehicle and System-Level Design Considerations to Facilitate the CEV Hardware-Software Integration, V&V, and Certification Processes," NASA Ames Research Center, Intelligent Systems Division, Final Report to JSC Task 3, October, 2006.
- ⁵⁷ARINC Specification 653-1, "Avionics Application Software Standard Interface," Airlines Electronic Engineering Committee, October 16, 2003.
- ⁵⁸Williams, D. K. and Neilan, P. J., "The Role of Safety Cases in Risk Management," European Convention on Security and Detection, Brighton, UK, May 1995.
- ⁵⁹Handbook, Preparation and Evaluation of Safety Cases, Bentham Technical Training Course, published by Balogh International, Inc., May 1994.