ISSN 1330-3651
UDC/UDK 656.7.053:004.942

FLIGHT CONTROL SYSTEM DEVELOPMENT USING SIMULATION – AN INTEGRATED APPROACH

Ivan D. Djokic, Zarko P. Barbaric

Original scientific paper

Flight control system development is a complex and multi-objective process, with extensive use of simulation. This paper suggests a new integrated approach in simulation usage – use of the same model throughout complete flight control system development process. Simulation starts with a refinement of system requirements and extends to preliminary design, design, hardware and software development, system integration, flying quality and reliability testing, and at the end, system validation and verification. The most difficult task is software reliability estimation and testing. A special mode of hardware-in-the-loop simulation configuration – "operational software reliability testing mode" is proposed for this task. The proposed simulation concept is applied on an unmanned aerial vehicle flight control system development, and simulation results of flying quality, reliability and vehicle clearance tests are shown as the example. The presented approach has proven to be a flexible tool for assessing flying qualities, hardware and software reliability and pilot-in-the-loop performance in a future simulated environment.

Keywords: *flight control system, hardware-in-the-loop, pilot-in-the-loop, real-time simulation*

Razvoj sustava za kontrolu leta primjenom simulacije – integrirani pristup

Izvorni znanstveni članak

Razvoj sustava za kontrolu leta je složen, mnogim ciljevima usmjeren proces, uz ekstenzivnu primjenu simulacije. U ovom se radu predlaže novi integrirani pristup pri primjeni simulacije – korištenje istog modela tijekom čitavog procesa razvoja sustava za kontrolu leta. Simulacija počinje usavršavanjem operabilnih potreba sustava i proteže se na pripremni projekt, projekt, razvoj hardvera i softvera, integraciju sustava, testiranje pouzdanosti i kvalitete letenja i, na kraju, validaciju i verifikaciju sustava. Najteži zadatak je procjena pouzdanosti softvera i ispitivanje. Specijalni tip konfiguracije simulacije hardware-in-the-loop – "provjera pouzdanosti operacionog softvera" – predlaže se za taj zadatak. Predloženi koncept simulacije primijenjen je na razvoj sustava za kontrolu leta bezposadne letjelice, a rezultati simulacije kvalitete leta, pouzdanosti i clearance testova letjelice daju se kao primjer. Opisani se pristup pokazao kao fleksibilan alat za procjenu letačkih sposobnosti, pouzdanosti hardvera i softvera i pilot-in-the-loop performance u budućem simuliranom okruženju.

Ključne riječi: *hardware-in-the-loop, pilot-in-the-loop, simulacija u stvarnom vremenu, sustav za kontrolu leta*

1 Introduction

Simulation is the technique by which a physical system can be represented mathematically by a computer program for the solution of a problem. Computing speed and software quality advances over the last 20 years have made flight simulation particularly effective in modeling the flight environment, and it is now an integral part of the aviation scene in the civil, military, manufacturing, and research fields. Aeronautical standards suggest performing piloted simulations during Flight Control System (FCS) development. As a minimum, the following simulations shall be accomplished: (a) piloted simulations using computer simulation of the FCS prior to hardware availability, and (b) piloted simulations using actual FCS hardware prior to first flight [1]. Compared to the flight environment, simulation gives close control of the conditions under investigation, and allows specific flight situations, some of which are rare or hazardous, to be available on demand. Compared to the use of aircraft for these activities, simulation causes no pollution, noise or other disturbance. For all but the simplest aircraft, flight simulation is also substantially less expensive than the use of the aircraft itself. Finally, simulators can be used at intensive rates of operation by day and night, and can carry out any exercise or function which is included in their data base irrespective of location, weather, and time of day or season of the year.

With rapid advancements in avionics systems, advanced cockpit controls, advanced cockpit displays, fly-by-wire technology and more, the need to move expediently from concept to certification is a fundamental requirement for a successful aircraft development project. The use of simulation to develop a superior aerospace or defense product has been accepted across the research institutes and industry. Simulation has reduced the weight of the aircraft, improved aerodynamic efficiency, extended aircraft range, and resulted in more reliable and efficient products.

For the sake of flying qualities, safety, reliability and mission effectiveness evaluation, authors propose in this work the integrated approach – use the same simulation model throughout complete FCS development process, and to combine simulation and optimization during every phase of aircraft development. This concept provides the operational profile for FCS in every phase of aircraft development and complete aircraft flight envelope, including search for the worst of dangerous cases of the flight control system. Different simulation configurations are proposed: all-digital, pilot-in-the-loop, hardware-in-the-loop simulation, and a special hardware-in-the-loop mode for operational software reliability testing. The flight control system development using simulation is illustrated by an example, the unmanned aerial vehicle (UAV).

2 Related works

The modern scientific simulation came with the development of analog computers in the 1930's; and progressed even further when the electronic digital computers were created. The very definition of an analog computer contains the notion of simulation, as a device which simulates some mathematical process and in which the results of this process can be observed as physical quantities, such as voltages and currents. While there is no doubt that the analog computer represents one aspect of simulation, the truly new simulation advances came with

the digital computer [2]. Real-time flight simulation of space vehicles presents special challenges to computing technology, in either man-piloted or automatically controlled space flights. At General Dynamics Astronautics, during 1960's, the problem was being solved through the joint use of a general purpose analog computer and a digital computer. This combination analog-to-digital and digital-to-analog converter is operated under digital computer program control. The complete closed loop operation, on the other hand, is under the control of the analog computer operator [3]. During 1970's a large hybrid computer simulation was used as an aid to design and develop the Digital Flight Control System (DFCS) and then used to validate the resulting DFCS airborne software. Flight control hardware components such as rate gyros, body mounted accelerometers, and hydraulic actuators were also used in the simulation [4]. Hybrid system models can greatly reduce the complexity of a physical system model by abstracting some of the continuous dynamics of the system into discrete dynamics. Hybrid system models are also useful for describing the interaction between physical processes and computational processes, such as in a digital feedback control system. One author presents a model of a complex control system that combines continuous-state physical system models with rich discrete-state software models in a disciplined fashion [5]. Some of the efforts are aimed to develop the computational methods for accurately determining static and dynamic stability and control characteristics of fighter and transport aircraft with various store configurations, as well as the aircraft response to pilot input. The main benefits of this effort are: 1) early discovery of complex aerodynamic phenomena that are typically only present in dynamic flight maneuvers and therefore not discovered until flight test, and 2) rapid generation of an accurate aerodynamic database to support aircraft and weapon certification by reducing required flight test hours and complementing current stability and control testing [6]. Some other works are focused on the study of how to evaluate the overall performance of flight control system based on general purpose computer system. By adopting hardware-in-the-loop simulation (HILS), a simulated real working environment can be constructed for flight control system, in which test and evaluation can be accomplished. Because of the strict real-time requirement of HILS, how to implement hard real-time capability in general personal computer systems must be analyzed systematically [7]. Before a new flight control system is released for flight, a huge number of simulations are evaluated to find weaknesses of the system. Clearance of a flight control system is a very important but time consuming process. It is impossible to cover all cases of the flight envelope and the pilot's commands using only simulation. There is also a need to consider uncertainties in the parameters of the simulation model and search for the worst combinations of uncertainties. One of the proposed ways is to combine simulation and optimization to search for the worst of dangerous cases of the flight control system [8].

Some of the simulation methods have focused primarily on analyzing and evaluating the performance of critical software, to assess current reliability and forecast future operability from observable failure data, using statistical inference techniques. However, none of these methods extends over the entire reliability process; most tend to focus only on failure observance during development, testing or operations. Discrete-event simulation has been used as an attractive alternative to analytical software reliability models, as it can capture a detailed system structure, and can be used to study the influence of different factors separately as well as in a combined fashion on dependability measures. The flexibility of discrete-event simulation to analyze complex systems was demonstrated [9]. Some authors have developed simulation procedures to describe and measure software reliability. They constructed and explained the debugging behavior through queue models. Using the proposed framework, the stochastic fault detection and correction processes were described under different conditions. The possible testing/debugging behavior of software system are simulated and studied based on real data [10]. For some critical applications, such as fault-tolerant flight control computer, simulation environment is used for the on-line monitoring. The simulation environment is designed to evaluate an improved on-line monitoring technique for processors with a built-in cache. This technique assumes that a monitor checks on-line whether the execution of a program is in accordance with the control flow graph created for the program off-line by a preprocessor [11]. NASA has developed and tested Intelligent Flight Control System (IFCS) that enhances control during a primary control surface failure or aerodynamic change due to a failure or modeling errors. Flight demonstration started early in the year 2006, on the NASA F-15 aircraft. The F-15 6-degree-of-freedom (6-dof) simulator was used in the evaluation test, comparing classic failure compensation with the neural network algorithm. The verification was very complex, and complexity posed by adaptive control systems primarily stems from the use of the learning algorithm. The flight test data were compared with the MATLAB/Simulink based validation and verification (V&V) tools developed under the IFCS Project. The V&V tools predictions are similar to those obtained from the flight test. This shows that V&V tools, based on simulation, have future application in the V&V of complex adaptive neural network based controllers [12]. Flight control system development using simulation, primarily for mission and safety reliability testing, was applied to the unmanned aerial vehicle project. Use of simulation is then extended on complete process, from requirements to validation and acceptance, but with slightly different models tailored for specific aims: aerodynamic optimization, definition of operational requirements, control law design, hardware and software design and development, system integration, system validation and verification, failure mode and effects analyses, and pilot training [13]. Similar approach was useful for an UAV helicopter development. Various hardware-in-the-loop experiments were conducted and approach was effective. For certain flight tests, the result of the hardware-in-the-loop simulation was able to provide safety alerts to human pilots in advance. As a result, the human pilot can overtake the automatic system and perform a timely manual control when the UAV helicopter becomes unstable [14].

3
Simulation concept

Aeronautical standards regulate that compliance with all FCS requirements shell be demonstrated through analysis. In addition, compliance with many of the requirements will be demonstrated by simulation, flight tests, or both [17]. Standard MIL-F-8785 also states that the danger, extent of difficulty of flight testing may dictate

simulation rather than flight test to evaluate some conditions and events, such as the influence of severe disturbances, events close to the ground, combined failure states and disturbances, etc. In addition, piloted simulation shall be performed before the first flight of a new airplane design in order to demonstrate suitability of the handling qualities, and also to demonstrate compliance with qualitative requirements in atmospheric disturbances and in the critical conditions.

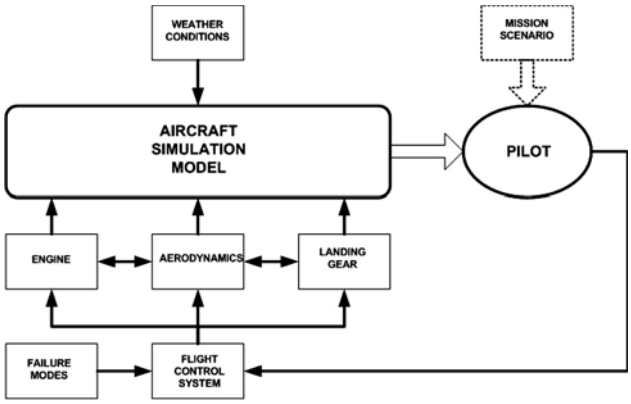

Figure 1 Aircraft simulation model

Related works show that many simulation models exist, with different levels of complexity, different accuracy, different interfaces and different applicability. Most of them are tailored for a specific problem and a specific method for problem solving. Having in mind aeronautical standards and interrelations between flying qualities, FCS failure modes, environmental disturbances, authors suggest the use of the same simulation model for different purposes during a FCS development process. The model suitable for such approach is shown in Fig. 1, where simulation modules are represented in the following manner:

Aerodynamics - Aircraft is presented as a rigid body with six-degree-of-freedom, three translations and three rotations. Six variables describing translation of the aircraft are velocity, angle of attack, sideslip angle and three center of gravity coordinates. Variables describing rotation of the aircraft are Euler angles (pitch, roll, and yaw) and three aircraft body angular rates. Conversion method of angular velocities to orientations in world coordinates is the Quaternion Method. Based on the unit sphere, the Quaternion Method provides an elegant method of defining rotations through the use of four parameters. Three of the coordinates describe the axis of rotation while the fourth is determined by the angle through which the rotation occurs. The aerodynamic coefficients are 14 nonlinear functions, obtained from wind tunnel. The flight envelope is limited by the aerodynamic coefficients obtained from the wind tunnel or flight tests.

Engine – Engine has different models, depending on the type (turboprop, turbojet, turbofan, etc.). Generally, turbojet engine can have two representations: (1) engine as a collection of subsystems (fan, compressor, combustor, turbine, mixer, nozzle), or (2) engine as a "total output", where outputs are complex functions of input parameters. Engine thrust is limited to the *X*-axis only, and no calculations are made for torque or gyroscopic effect since these are second order effects for high performance aircraft.

Landing gear – Landing gear is described by resulting forces and moments, and the model includes the effects of linkage dynamics, damper characteristics, friction, and tire nonlinearities.

Flight Control System – Flight control systems can get different forms – mechanical, analog, digital, hybrid, simplex, redundant, fault-tolerant, depending on particular aircraft and design phase.

Weather Conditions – Atmospheric disturbance model can be used in complete flight envelope. The effects of wind shear, turbulence and gust may be analyzed separately or in a combination. Some analyses and piloted simulation are required considering complete environmental representation, demonstrating compliance with the requirements with the cumulative effects of wind shear, turbulence and gusts. Two models of disturbance can be used - von Karman form shall be used for the continuous turbulence model, so that the flying qualities will be consistent with the comparable structural analyses. When it is not feasible to use the von Karman form, use of the Dryden form is also possible. In general, both the continuous turbulence model and the discrete gust model shall be used [17]. During flight simulation, the designer or instructor should be able to set up a specific weather which can help him to fly the aircraft in different scenarios. In weather environment, the following has to be simulated: 1) the effects of three-dimensional wind shear, and 2) turbulence. The effects of turbulence are calculated in the mathematical mode1 of the simulated aircraft and introduced through the flight equations.

Pilot – Designing and performing sets of handling qualities can be done using simulation model of a pilot at an early stage of the project. In all-digital simulation the human equalization characteristics are best defined by description of a typical pilot model. One of the most widely applied models is used in the proposed simulation concept, the so-called Precision Pilot Model (PPM). Pilot is described by the transfer function (1).

$$Y_{\mathrm{P}}(s) = K_{\mathrm{P}} \cdot \left(\frac{1 + T_{\mathrm{L}} \cdot s}{1 + T_{\mathrm{I}} \cdot s} \right) \cdot \left(\frac{\mathrm{e}^{-ts}}{1 + T_{\mathrm{N}} \cdot s} \right), \tag{1}$$

where K_{P} is the 'Pilot Gain' representing the pilot's ability to respond to an error in the magnitude of a controlled variable, T_{L} is the 'Lead Time Constant' reflecting the pilot's ability to predict a control input and T_{I} is the 'Lag Time Constant' which describes the agility of the pilot and ease with which the pilot generates the required input. These three parameters are known collectively as the 'human equalization characteristics'. The remaining two terms within the transfer function can be defined as the 'inherent human limitations' where e^{-ts} represents a pure time delay describing the period between the decision to change a control input and the change starting to occur. Finally T_{N} is the 'neuromuscular lag time constant' which represents the time constant associated with contraction of the muscles through which the control input is applied by the pilot. During piloted simulations this simulation module is inactive.

Mission scenario – The FCS designer or instructor can define a flight scenario to the pilot or pilot's simulation model. The mission scenario may vary from a very simple (one step input, for example), up to a very complicated combat mission. The mission scenario depends on FCS feature examined through the current test.

Failure modes – Standard for flying qualities of piloted aircrafts, MIL-F-8785, makes relationship between flying qualities and failure status of aircraft. When failure status exists, degradation in flying qualities is permitted only if

the probability of encountering failure is sufficiently small. Also, MIL-F-8785 determines levels of flying qualities in different atmospheric disturbances: 1) light to calm disturbances – *Satisfactory* qualitative flying quality, 2) moderate to light disturbances – *Acceptable* or better flying quality, and 3) severe to moderate disturbances – *Controllable* or better flying quality. This means that every potential FCS failure should be examined in simulated environment, similar to the real environment. The proposed simulation concept allows the generation of FCS failure modes and continuous monitoring of flying qualities. Also, aircraft motions following sudden flight control system or component failures are monitored and dangerous conditions are registered.

The proposed simulation concept can be used scalable, with three basic configurations: A. all-digital simulation, B. pilot-in-the-loop simulation, and C. hardware-in-the-loop simulation.

A. All-digital simulation

When the control inputs to the system can be predetermined and are programmable, all-digital simulation is possible. This kind of simulation is suitable for the early phase of system development, where many parameters of the system are variable. Here the running time of the computer program is not related to the real world time.

B. Pilot-in-the-loop simulation

In the event that the control inputs necessary for the testing procedure are dynamic in nature, or cannot be predetermined, such as the pilot response in an actual aircraft, the term simulation takes a new dimension known as real-time pilot-in-the-loop simulation. This new dimension calls for strict correspondence between the computer time and the real world time. Inputs and outputs to hardware devices must be synchronized to a real-time clock and cannot be time-scaled as in the all-digital computing environment. A real-time simulation of the aerial vehicle corresponds to an actual vehicle flight as viewed by an observer. When an actual piece of flight equipment, such as a control device, is placed in the simulation, then the simulation becomes hardware-in-the-loop with the same characteristics as pilot-in-the-loop.

C. Hardware-in-the-loop simulation

Embedded systems are designed to control complex plants such as UAVs, aircrafts, weapon systems, and jet engines. They generally require a high level of complexity within the embedded system to manage the complexity of the plant under control. Hardware-in-the-loop (HIL) simulation is a technique that is used increasingly in the development and test of complex real-time embedded systems. The HIL simulation also includes electrical emulation of sensors and actuators. These electrical emulations act as the interface between the plant simulation and the embedded system under test. The value of each electrically emulated sensor is controlled by the plant simulation and is read by the embedded system under test. Likewise, the embedded system under test implements its control algorithms by outputting actuator control signals. Changes in the control signals result in changes to variable values in the plant simulation. The HIL simulation is an effective tool for development of highly reliable systems, influencing development and test efficiency (cost, duration, and risk).

Hardware-in-the-loop simulation mode for operational software reliability testing

Two reliability requirements are defined for flight control systems: a) mission accomplishment reliability, and b) safety reliability [1]. Mission accomplishment reliability is specified by relation (2).

$$Q_{\mathrm{M}}(\mathrm{FCS}) \le (1 - R_{\mathrm{M}}) \cdot A_{\mathrm{M}}(\mathrm{FCS}), \tag{2}$$

where $Q_{\mathrm{M}}(\mathrm{FCS})$ is maximum mission unreliability due to relevant FCS material failures, R_{M} is specified overall aircraft mission accomplishment reliability and $A_{\mathrm{M}}(\mathrm{FCS})$ is mission accomplishment allocation factor for flight control system. The probability of aircraft loss per flight defined as extremely remote, due to relevant material failures in the flight control system, shall not exceed value defined by relation (3).

$$Q_{\mathrm{S}}(\mathrm{FCS}) \le (1 - R_{\mathrm{S}}) \cdot A_{\mathrm{S}}(\mathrm{FCS}), \tag{3}$$

where Q_{S} is maximum acceptable aircraft loss rate due to relevant FCS material failures, $A_{\mathrm{S}}(\mathrm{FCS})$ is flight safety allocation factor for flight control system, and R_{S} is overall aircraft flight safety requirements. When FCS is a digital system, then allocation factors $A_{\mathrm{M}}(\mathrm{FCS})$ and $A_{\mathrm{S}}(\mathrm{FCS})$ include operational flight control software. This means that operational software mission reliability and safety reliability must be verified. The simulation concept proposed in this paper is suitable for software reliability testing, using a special hardware-in-the-loop mode.

The fundamental assumption of reliability-process simulation is that every stochastic event results from an underlying, instantaneous *conditional event-rate random process*. A conditional event-rate process is one for which the probability that an event occurs in the certain interval (t, $t + \Delta t$), given that it has not occurred prior to time t, is equal to $\beta(t) \cdot \Delta t$ for some function $\beta(t)$. The Δt must always be chosen such that the variations in the failure rate $\beta(t)$ over the incremental time intervals (t, $t + \Delta t$) are negligible, and such that $\beta(t) \cdot \Delta t < 1$, so that instantaneous event probability does not reach unity.

Some published analytic models treat or approximate the overall software reliability growth as a nonhomogeneous Poisson process in execution time. Many of these differ only in the forms of their rate functions. For example:

- The Jelinski-Moranda model [15] is credited with being the first reliability model. It belongs to a class of exponential order statistic model that assumes that fault detection and correction begins when a program contains n_0 (unknown) faults and all the faults have the same per-fault failure rate φ. The model assumptions are: 1) at the beginning of testing, there are n_0 faults in the software code with n_0 being an unknown, 2) each fault is equally dangerous with respect to the probability of its instantaneously causing a failure, and the hazard rate of each fault does not change over time, but remains constant at φ, 3) the failures are not correlated, i.e. given n_0 and φ the times between failures ($\Delta t_1, \Delta t_2, ..., \Delta t_{n0}$) are independent, 4) whenever a failure has occurred, the fault that caused it is removed instantaneously, and 5) the software is operated in a similar manner as that in which reliability predictions are to be made. As a consequence of these assumptions,

the program hazard rate after removal of the $(i-1)^{th}$ fault z is proportional to the number of faults remaining in the software, with hazard rate of one fault $z_a(t) = \varphi$, being the constant of proportionality:

$$z(\Delta t|t_{i-1}) = \varphi[n_0 - M(t_{i-1})] = \varphi[n_0 - (i-1)], \quad (4)$$

where $M(t_{i-1})$ is the number of faults experienced by time t_{i-1}. From the model assumptions, one can determine software reliability if the time-between-failure occurrences are known $\Delta t_i = t_i - t_{i-1}$, $i=1, 2, \ldots, n$.

- The Goel-Okumoto model [16] is based on the following assumptions: 1) the number of failures experienced by time t follows a Poisson distribution with mean value function $\mu(t)$, 2) the number of software failures that occur in $(t, t+\Delta t]$ with $\Delta t \to 0$ is proportional to the expected number of undetected faults, $n_0 - \mu(t)$ and the constant of proportionality is φ, 3) for any finite collection of times $t_1 < t_2 < \ldots < t_n$ the number of failures occurring in each of the disjoint intervals $(0,t_1)$, (t_1,t_2), …, (t_{n-1},t_n) is independent, 4) whenever a failure has occurred, the fault that caused it is removed instantaneously and without introducing any new fault into the software. Since each fault is perfectly repaired after it has caused a failure, the number of inherent faults in the software at the beginning of testing is equal to the number of failures that will have occurred after an infinite amount of testing. The Goel-Okumoto model deals with overall reliability growth, in which the failure intensity function is given by relation (5).

$$\lambda(t) = n_0 \cdot \varphi \cdot e^{-\varphi t}, \quad (5)$$

where n_0 (number of initial faults) and φ (per-fault failure rate) are constant parameters.

From the examples it is clear that simulation must provide software operational environment, similar to real world environment, and statistics of mean time between failures occurring during simulation. Many other existing models require the same data. The most difficult task is to detect a software failure, especially when a flight control system has embedded a fault-tolerant mechanism. The proposed simulation concept has a special hardware-in-the-loop simulation mode, shown in Fig. 2, where *the automatic failure detection logic* is used.

Authors have implemented the automatic failure detection logic with the basic idea to monitor and compare behaviors of two vehicle dynamics – one driven by "fault-free software" and the other driven by "operational flight control software" (which is under the reliability test). The simulated aircraft transients are the aircraft motions due to pilot commands, transfer to alternate control modes, disturbances and failures. The only difference can exist in flight control software (fault-free and operational), and different transients can be produced by a failure in one of them. Authors believe that fault-free software also has undetected faults, but its reliability is much higher than the reliability of prototype embedded software. This assumption is based on a reliable high level language used for requirement definition, coding and debugging of fault-free software, as well as on a longer software test time resulting in high cumulative error detection. The more

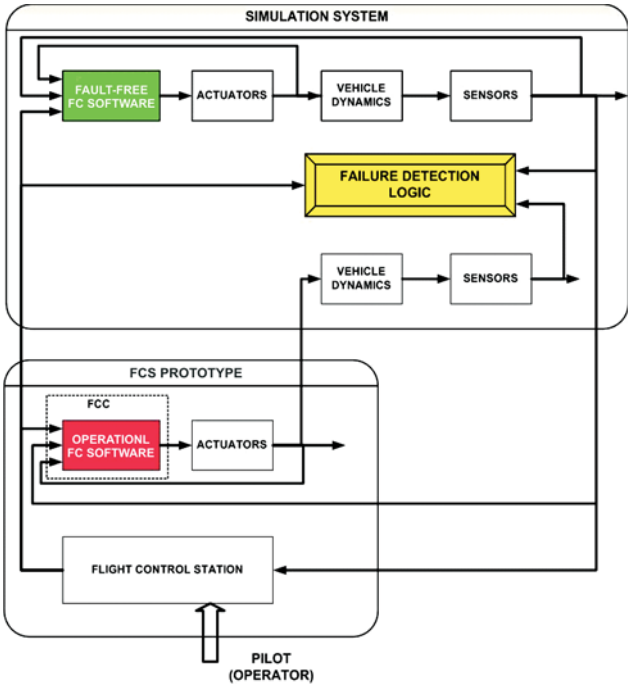

Figure 2 Hardware-in-the-loop simulation mode for operational software reliability testing

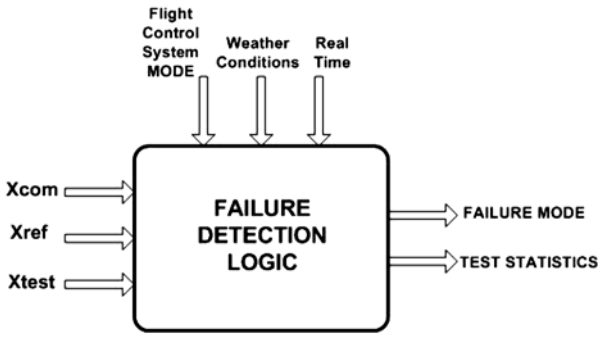

Figure 3 Failure detection logic of a flight control system

detailed presentation of the automatic failure detection logic is shown in Fig. 3.

In order to detect a failure, detection logic monitors: FCS modes, weather conditions, pilot inputs (Xcom), outputs of aircraft model commanded by the fault-free software (Xref) and outputs of aircraft model commanded by the operational software (Xtest). The failure detection logic compares Xref and Xtest and when differences are out of certain thresholds, illustrated by Fig. 4, a failure is declared.

The vectors Xref and Xtest are composed of pitch angle, bank angle, course, normal acceleration, lateral acceleration, and roll rate. The automatic failure detection logic has predefined thresholds, based on standard MIL-F_9490D [1], where acceptable failure transients are defined for different flight control modes. Fig. 4 shows a part of failure detection logic, dedicated for the aircraft normal acceleration transients.

The logic thresholds are based on the following: a) the transient motions resulting from intentional engagement or disengagement of any portion of the flight control system by the pilot shall be such that dangerous flying qualities never result, b) with pilot controls free, transients resulting from intentional engagement or disengagement shall not exceed $0{,}1 \cdot g$ normal acceleration within the operational flight envelope for two seconds following transfer, and c) in atmospheric disturbances the same degradation of flying

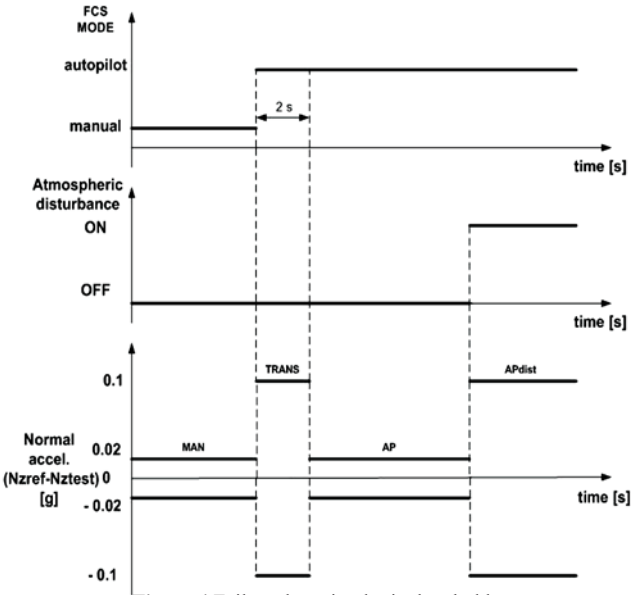

Figure 4 Failure detection logic thresholds

qualities is permitted. Based on this assumptions failure detection thresholds are predefined for every specific project. Thresholds shown by Fig. 4 are from an unmanned aerial vehicle project, where flight control system has two operational modes, manual (pilot generates control commands) and autopilot (autopilot system generates control commands). The logic recognizes five different system states – manual (MAN), engagement transient (TRANS), autopilot (AP), atmospheric disturbances in manual and autopilot modes (MANdist, APdist). MANdist and APdist thresholds are the same. When vehicle transients are out of predefined limits a failure is declared and simulation stopped. Outputs from the failure detection logic are failure code (unique identification of stopping cause) and simulation statistics (number of successful executions of operational software before failure). Simulation statistics gives the opportunity to apply a software reliability model and to estimate the operational software reliability.

4
Example – The Unmanned Aerial Vehicle FCS

The UAV flight control system was developed using simulation concept described in this paper. The vehicle is shown in Fig. 5. The flight control system is digital, with two normal modes of operation (autopilot and manual) and one emergency mode. Required safety reliability for 5 hours endurance of flight is 0,98. System is simplex, without fault-tolerance.

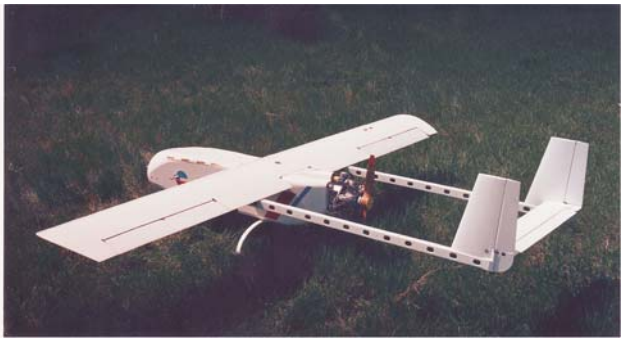

Figure 5 Unmanned aerial vehicle

Flying quality tests – Many vehicle configurations have been analyzed using all-digital simulation (different aerodynamic configurations, different payloads, and different control laws). Figure 6 shows an example of all-digital simulation – UAV pitch response during examination of phugoid stability for one vehicle configuration and one flight condition.

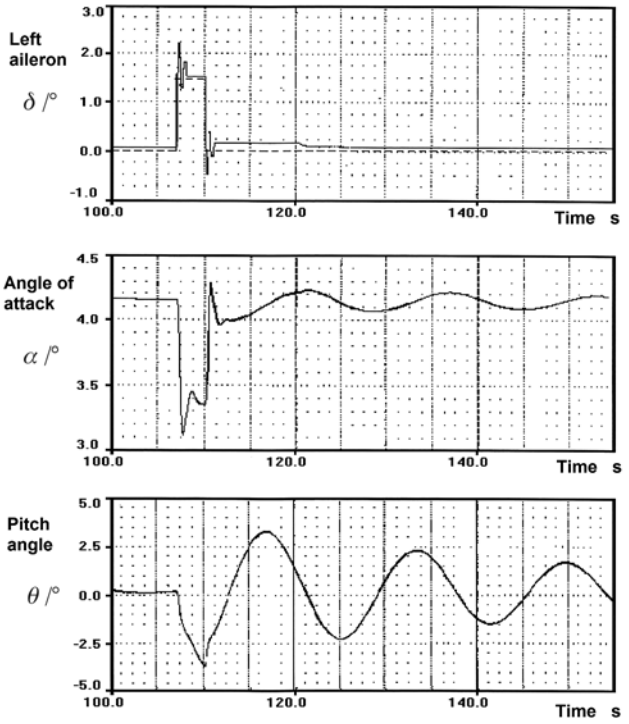

Figure 6 UAV pitch response (all-digital simulation)

The phugoid stability requires dumping coefficient for the long-period longitudinal oscillations, for flying quality level 1, to be at least ζ_P=0,04. Fig. 6 shows that the disturbance is made by aileron input (δ) and after that command is free, angle of attack (α) oscillates near a stable steady state and long-period oscillations of pitch angle (θ) have ζ_P better than 0,04.

Manual and autopilot mode have been tested with different pilots in different flight conditions using pilot-in-the-loop simulation. From landing statistics it was estimated that manual mode of operation is unacceptable for landing under moderate and severe weather conditions. Fig. 7 shows one of the results, landing trajectory with untrained operator in the loop, under turbulent weather conditions.

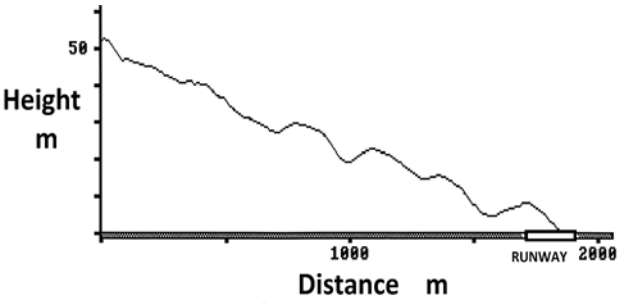

Figure 7 UAV landing trajectory (pilot-in-the-loop simulation)

Trajectory has unacceptable variations in height during approach and landing point is unacceptable far from the touchdown zone (first 100 m of the runway).

Reliability tests – Hardware-in-the-loop has been used intensively for flying qualities and system reliability assessment. Total simulation time during acceptance test in this configuration was 693 hours. Four simulation tests were stopped by safety noncritical errors (calculation accuracy) and decision was made not to change operational software. One safety critical fault was discovered, aero-servo-elastic instability, caused by vehicle-FCS elastic coupling. Fig. 8 shows result of the hardware-in-the-loop simulation – left aileron and rudder deflections, with aileron oscillations as a consequence of the discovered instability.

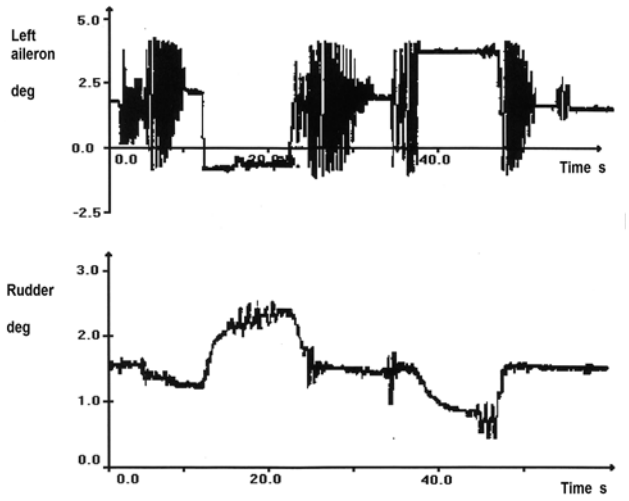

Figure 8 UAV aero-servo-elastic instability (hardware-in-the-loop simulation)

The fault is removed by a low-pass filter redesign and with a new operational software code. After the fault removal and new functional and reliability testing required system reliability was confirmed (R_S= 0,995).

The UAV clearance - The flight control system must also pass several clearance criteria to be certified as cleared. The basic aim of the clearance process is to search for possible departures of the aircraft. Loss of stability or controllability, or both, is termed as an aircraft departure. Departure resistance testing is one of the most difficult tasks to accomplish when testing highly nonlinear systems. In the UAV project clearance criteria are studied, especially departure criterion based on the angle of attack and angle of sideslip. The criterion describes the possible loss of stability, since the UAV will operate at different weather conditions and with operators with different experiences, which can lead to dangerous flight conditions (high angle of attack and sideslip). Fig. 8 shows UAV landing points for different weather conditions: cross-wind 1 m/s, cross-wind 5 m/s and moderate turbulence. Based on the simulation results UAV take-off and landing was limited to moderate weather conditions (cross-wind ≤ 2,5 m/s and moderate turbulence).

The simulation results show that 50 % of landing points are far from the touchdown zone when a disturbance is cross-wind 1 m/s, 60 % are far from the touchdown zone when a disturbance is cross-wind 5 m/s, and only one landing is successful (in touchdown zone) when a disturbance is moderate turbulence. Based on the complete set of simulation results, UAV take-off and landing was limited to moderate weather conditions (cross-wind ≤ 2,5 m/s and moderate turbulence), and autopilot mode was declared as the basic FCS mode.

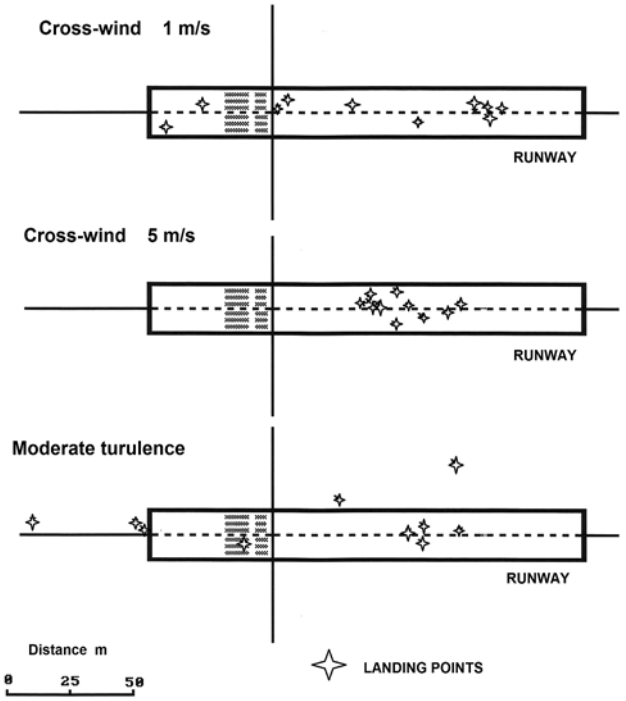

Figure 9 UAV landing points under different weather conditions (hardware-in-the-loop simulation)

The use of hardware-in-the-loop simulations for dynamic system verification and flight qualification is generally required for mission and safety critical software. The UAV project approved the reason for using hardware-in-the-loop simulations, providing greater accuracy and realism. Without detailed modeling of the flight computers and systems, it is impossible to evaluate the interaction of their embedded systems. Using the flight hardware and software in a dynamic, closed-loop simulation lets you realistically and cost effectively test features that stand-alone software or systems testing cannot test. These features include subsystem interaction, system communication delays, real-time performance, operational distribution of software inputs (essential for software reliability assessment), and failure modes. UAV hardware-in-the-loop simulation, especially with the reliability testing mode, proved all benefits of the integrated approach in the use of simulation during a flight control system development.

5
Conclusion and Suggestions

The new integrated approach in simulation usage has been given. The same simulation model is used throughout the complete flight control system development process. Three simulation configurations are used: all-digital, pilot-in-the-loop and hardware-in-the-loop. The special hardware-in-the-loop mode, with the original automatic failure detection logic for software reliability testing, is presented. The proposed simulation concept is applied to the unmanned aerial vehicle flight control system development. A part of simulation results, illustrating flying quality tests, reliability validation and verification, and vehicle clearance tests are shown. The presented approach has proven to be a flexible tool for assessing flying qualities, hardware and software reliability and pilot-in-the-loop performance in a future simulated environment. The final result is the efficient use of the proposed integrated simulation approach in a flight control system development with minimized cost, time and risk.

Further simulation enhancements are expected in integrating and combining different simulated components, designed using one or more modeling packages and written in one or more code languages (Ada, C, C++, Fortran), and performing a combined simulation of these multiple components. During this new kind of simulation all models are executed in a synchronized manner using a platform like workstation or PC.

Acknowledgement

This work was partially supported by the Ministry of Education and Science of the Republic of Serbia under Grant TR-32023.

6 References

[1] MIL-DTL-9490E, Flight Control Systems – Design, Installation and Test of Piloted Aircraft, General Specification for, Department of Defense, 2008, superseding MIL-F-9490D (USAF), Department of Defense, 1975
[2] Harman, H. H. Simulation: A Survey, From the collection of the Computer History Museum, URL: http://www.computerhistory.org,
[3] Wilson, A. N. Use of a combined analog-digital system for Re-entry Vehicle Flight Simulation, From the collection of the Computer History Museum, URL: http://www.computerhistory.org
[4] Jackson, R. S.; Bravdica, S. A. Software validation of the Titan IIIC digital light control system utilizing a hybrid computer, From the collection of the Computer History Museum, URL: http://www.computerhistory.org
[5] Neuendorffer, S. Modeling Real-World Control Systems: Beyond Hybrid Systems, // Proceedings of the 2004 Winter Simulation Conference/ R. G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A. Peters, eds., Washington, 2004, 240-248.
[6] Morton, S. A.; Forsythe, J. R.; McDaniel, D. R.; Bergeron, K.; Cummings, R. M.; Goertz, S.; Seidel, J.; Squires, K. D. High Resolution Simulation of Full Aircraft Control at Flight Reynolds Numbers, HPCMP users group conference, Pittsburgh, 2007, URL: http://www.dtic.mil/dtic/tr/fulltext/u2/p023734.pdf
[7] Xun, W.; Shu-Xing, Y.; Lei, Z. Dynamic Test and Evaluating System for Flight Control System// 2008 ISECS International Colloquium on Computing, Communication, Control, and Management, Guangzhou, China, Vol. 02, pp. 189-192, 2008
[8] Fredman, K.; Freiholtz, A. Use of Simulation Optimization for Clearance of Flight Control Laws, Department of Electrical Engineering, Linkoping University, Sweden, 2006
[9] Gokhale, S. S.; Lyu, M. R.; Trivedi, K. S. Reliability Simulation of Component-Based Software Systems. // ISSRE'98 Proceedings of the Ninth International Symposium on Software Reliability Engineering, Publisher IEE Computer Society Washington, USA, pp. 192- 201, 1998
[10] Lin, C-T.; Huang, C-Y.; Sue, C-C. Measuring and Assessing Software Reliability Growth through Simulation-Based Approaches. // Proccedings 31st Annual International Computer Software and Applications Conference (COMPSAC 2007), Vol. 1, pp. 439-448
[11] Punt, M.; Djordjevic, J.; Tomasevic, M. A Simulation Environment for the On-Line Monitoring of a Fault Tolerant Flight Control Computer// Proceedings of First IEEE Eastern European Conference on the Engineering of Computer Based Systems, Publisher IEEE Computer Society Washington, DC, USA, pp. 100-109, 2009
[12] Soares, F.; Burken, J. A Flight Test Demonstration of On-line Neural Network Applications in Advanced Aircraft Flight Control System// International Conference on Computational Intelligence for Modeling Control and Automation, and International Conference on Intelligent Agents, Web Technologies and Internet Commerce (CIMCA-IAWTIC'06), URL: http://doi.ieeecomputersociety.org/10.1109/CIMCA.2006.6
[13] Djokic, I. D. Design of Digital Flight Control System with Highly Reliable Operational Software, PhD thesis, University of Belgrade, (ETF), 1996.
[14] Cai, G.; Chen, B. M.; Lee, T. H.; Dong, M. Design and implementation of a hardware-in-the-loop simulation system for small-scale UAV helicopters, Mechatronics, Vol. 19, (2009), pp. 1057–1066.
[15] Jelinski, Z.; Moranda, P. B. Software reliability Research, in Statistical Computer Performance Evaluation, edited by W. Freiberger, 1996., pp. 465-484.
[16] Goel, A. L.; Okumoto, K. Time-dependant Error-Detection Rate Model for Software Reliability Measurement, IEEE Trans. Reliability, R-28, 3(1979), pp. 206-211.
[17] MIL-F-8785, Flying Qualities of Piloted Airplanes, Department of Defense, USA, 1980.

Authors' addresses

Ivan Djokic
State University Novi Pazar
Vuka Karadzica bb
36300 Novi Pazar, Serbia
E-mail: idjokic@np.ac.rs

Zarko Barbaric
State University Novi Pazar
Vuka Karadzica bb
36300 Novi Pazar, Serbia
E-mail: zbarbaric@np.ac.rs