ON THE USE OF BEHAVIORAL MODELS FOR THE INTEGRATED PERFORMANCE AND RELIABILITY EVALUATION OF FAULT-TOLERANT AVIONICS SYSTEMS

Alejandro D. Domínguez-García, John G. Kassakian, Joel E. Schindall
Laboratory for Electromagnetic and Electronic Systems, Massachusetts Institute of Technology, Cambridge, MA

Jeffrey J. Zinchuk
Systems Engineering and Evaluation, The Charles Stark Draper Laboratory, Inc., Cambridge, MA

***Abstract*— In this paper, we propose an integrated methodology for the reliability and performance analysis of fault-tolerant systems. This methodology uses a behavioral model of the system dynamics, similar to the ones used by control engineers when designing the control system, but incorporates additional artifacts to model the failure behavior of the system components. These artifacts include component failure modes (and associated failure rates) and how those failure modes affect the dynamic behavior of the component. The methodology bases the system evaluation on the analysis of the dynamics of the different configurations the system can reach after component failures occur. For each of the possible system configurations, a performance evaluation of its dynamic behavior is carried out to check whether its properties, e.g., accuracy, overshoot, or settling time, which are called performance metrics, meet system requirements. After all system configurations have been evaluated, the values of the performance metrics for each configuration and the probabilities of going from the nominal configuration (no component failures) to any other configuration are merged into a set of probabilistic measures of performance. To illustrate the methodology, and to introduce a tool that we developed in MATLAB/SIMULINK® that supports this methodology, we present a case-study of a lateral-directional flight control system for a fighter aircraft.**

I. INTRODUCTION

Fault-tolerant systems that are required for aircraft avionics need a thorough understanding of their behavior, and require powerful evaluation tools during their development phase, in order to fully understand and quantify potential component or subsystem failures and design proper failure recovery mechanisms.

There are in the literature [1], [2] well-developed techniques in the field of control engineering to design effective and robust control systems and to analyze their performance in the presence of disturbances and model uncertainties. Applications of these techniques to the design of aircraft avionics can be found in the literature [3]. The application of these techniques relies on having a behavioral model of the system dynamics, which is usually a set of differential equations, but can be a more general mathematical model.

The analysis of system behavior in the presence of components or subsystem failures and therefore system reliability, availability, and safety, is usually carried out using a qualitative description of the system's functionality. It describes how components and subsystems are interconnected to fulfill the functions they are designed for, and how component failures can propagate to other components through their interconnections and affect the system functionality [4], [5], [6]. For conventional systems, this approach is valid, as it is possible to evaluate how a system can fail to perform the function for which it was designed without paying attention to the dynamics. However, this is not the case for large complex systems or embedded software-intensive systems, which are characteristic of fault-tolerant systems [7]. For these kinds of systems, it is very difficult, if not impossible, to understand how the system behaves in the presence of hardware failures or software malfunctions without using a behavioral model of the system to quantify its performance under failure conditions.

In this paper, we propose a methodology that uses a behavioral model of the system dynamics, similar to the ones used by control engineers when designing the control system, but with additional features to model component failure behaviors. These features include component failure modes (and associated failure rates) and how these failure modes affect the dynamic behavior of the component. Starting from

1-4244-0378-2/06/$20.00

the nominal configuration, the system will evolve to different *system configurations*, depending on which components failed, how these components failed, and the order in which the components failed (when more than one failed). For each of the possible system configurations, a *performance evaluation* of the system dynamic behavior is carried out to check whether some of its properties, called *performance metrics*, meet system operational requirements. After all system configurations have been evaluated, the values of the performance metrics for each configuration and the probabilities of going form the nominal configuration to any other configuration are merged into a set of *probabilistic measures of performance*. The proposed methodology allows one to assess not only system reliability and safety, but other important dynamic performance metrics that might be relevant in the design of a fault-tolerant system. The behavioral model not only includes the system architecture (the avionics) under evaluation, but it also includes a model of the physical system to be controlled (the aircraft). The performance evaluation is based on how the physical system behaves when failures in components within the control system occur.

The idea of using a dynamic model of the system under control was proposed first in the nuclear engineering field by Amendola in 1981 to study the likelihood of accident sequences in a nuclear reactor, referred to as the Event Sequences and Consequences Spectrum (ESCS) [8]. In 1987, Aldemir [9] formalized some of the ideas introduced in [8], and introduced the idea of using a Markov model to compute the likelihood of different accident sequences in the reactor. In 1992 Devooght and Smidts laid down a rigorous mathematical formulation of the problem [10]. This methodology is commonly referred in the nuclear engineering field as Dynamic Probabilistic Risk Assessment (DPRA).

The methodology proposed in this paper is similar to DPRA in the sense that it makes use of a dynamic (behavioral) system model, with additional information to model component behavior. It also makes use of a Markov chain to model the stochastic transitions between system configurations that take place when components fail. However, there are certain aspects of the kind of problems that DPRA tries to solve that makes the mathematical formulation of DPRA much more complex than the formulation of our methodology. In aerospace problems the dynamic time constants are on the order of seconds. Therefore, when a component failure occurs, the system either recovers within a short transient period, in the order of seconds, reaching a new stable steady-state (associated with a new system configuration), or it quickly becomes unstable. In this scenario, the likelihood of two components failing within a time on the order of magnitude of the system dynamic time constants is negligible relative to the likelihood of just one component failing. Therefore, the stochastic transitions among configurations can be modeled as a process independent of the system dynamics, and the resulting model is formulated in terms of just the probability of the system configurations at a given time. This is not the case in nuclear power plants, where the decoupling of the stochastic transitions between configurations and system state dynamic variables cannot be done [11]. This is due to the fact that in nuclear power plants the time constants of the transitions from one configuration to another are large enough that the likelihood of another failure taking place before the system reaches a new steady-state is relevant. Therefore, in DPRA the stochastic model is formulated in terms of the probability of the system dynamic variables in a given configuration at a given time.

Section II of this paper presents and explains in detail the elements of the methodology and lays out its mathematical foundations. Section III presents the main features of a MATLAB/SIMULINK® tool that supports the proposed methodology. In Section IV a lateral-directional flight control system for a fighter aircraft is presented as a case-study. Concluding remarks and future work are presented in Section V. Additionally, all the dynamic models and parameters of the components for the analyzed case-study presented in Section IV are collected in several appendices at the end of the paper. The notation used is listed at the end of the paper as well.

II. Methodology

The methodology uses a model of the system dynamics plus additional features to model component failure behavior. These features include component failure modes (and associated failure rates) and how these failure modes affect the dynamic behavior of the component. Depending on which components failed, how this components failed, and the order in which components failed (when more than one failed), the system will evolve from the nominal configuration (no failures) to other configurations with different

dynamics. Each possible system configuration will be analyzed to check whether its dynamic properties, called *performance metrics*, meet system operational requirements. Once all system configurations have been evaluated, the performance metrics for each configuration and the probabilities of going from the nominal configuration to any other configuration are merged into a set of *probabilistic measures of performance*.

A. System Configurations

The system is composed of different interconnected components. When there are no failures in the system, the system is said to be in its nominal configuration, and its dynamics can be described by

$$\begin{aligned} \dot{x}(t) &= f_i(x(t), u(t)) \\ y(t) &= g_i(x(t), u(t)) \end{aligned} \tag{1}$$

where the subindex i equals 1 for the nominal configuration. The system can evolve from the nominal configuration (no component failures) to other dynamic configurations depending on the status of the components within the system. These statuses are dictated by the component failure modes. Each system configuration in the presence of component failures, $i = 2 \dots N$, can also be described by (1).

The transitions between configurations occur stochastically and are triggered by random failures within the system components. Each component can have different failure modes, and each failure mode has an associated failure rate. The system will be in any of the configurations, i , within system *global evaluation time* T (system life-span, preventive maintenance period or mission time), with certain probability $p_i(T)$ given that the system was in the configuration $i = 1$ at $T = 0$. We assume that the system evolves between configurations in a Markovian fashion, i.e., the system configurations at future times will only depend on the configuration at the present time. Therefore the Chapman-Kolmogorov equations can be used to compute the system configuration probabilities as

$$\frac{dp_k(t)}{dt} = -\lambda_k(t)p_k(t) + \Sigma_{i \neq k}\lambda_{ik}(t)p_i(t) \tag{2}$$

where $\lambda_k(t)$ results from adding all the individual failure rates associated with the different component failure modes that can trigger a system transition out of configuration k. $\lambda_{ik}(t)$ is the failure rate associated with the failure mode that triggers the transition from system configuration i to system configuration k. As mentioned in the introduction, and shown here, the formulation of the stochastic model of the transitions between configurations is independent of the system dynamic variables, and, therefore, it is possible to compute the configuration probabilities $p_k(t)$ independently of the system dynamics state-variables x. DPRA tries to compute the probability of the system dynamic variables in a given configuration at a given time $p_k(x, t)$, and therefore the formulation of the Chapman-Kolmogorov equations becomes much more complicated [10].

B. Performance Metrics

Performance metrics $Z_1, Z_2, \dots Z_m$ are measurable properties of the system that will quantify how well it performs the functions for which it was designed. Performance metrics can be system functionality related, e.g., in a tracking-system, relevant performance metrics are the tracking error, the overshoot or the settling time. However, non-functional metrics can be considered as well, e.g., in the tracking system, the power consumption may be important.

For each performance metric Z_i with $i = 1 \dots m$, a set of requirements must be defined. These requirements can be given in the form of bounds as

$$\begin{aligned} L_1 < Z_1 < U_1 \\ L_2 < Z_2 < U_2 \\ \dots \\ L_m < Z_m < U_m. \end{aligned} \tag{3}$$

C. Performance Evaluation and Probabilistic Measures of Performance

For each system configuration represented by (1), the behavior of the system is analyzed in order to quantify its performance metrics. For a given system configuration, $\{f_i, g_i\}$, if any of the performance metrics $Z_1, Z_2, \dots Z_m$, do not meet the specified requirements given in the form of bounds by (3), then that configuration is regarded as a system failure.

The performance metrics $Z_1, Z_2, \dots Z_m$ are useful in determining whether each possible system configuration is declared as *failed* or *non-failed*, but it is necessary to define another set of measures to quantify the system as a whole, i.e., aggregated measures for all the possible system configurations. This set of measures, called probabilistic measures of performance, will be

helpful in identifying the weakest points in the design and therefore improving the system design as a whole. System reliability is an example of an aggregated measure of performance when the performance metric considers only whether the system is functional or not.

The two elements that compose the probabilistic measures of performance are

1) the system configuration probabilities $p_i(T)$ given by (2), and
2) a reward model, given by

$$\begin{aligned} r_1 &= h_1(Z_1) \\ r_2 &= h_2(Z_2) \\ &\dots \\ r_m &= h_m(Z_m) \end{aligned} \tag{4}$$

where $h_1(\cdot), h_2(\cdot), \dots, h_m(\cdot)$ are real functions.

The probabilistic measures of performance are computed as the expectation of the reward model by

$$\begin{aligned} \mathbb{E}(R_1) &= \textstyle\sum_k h_1(z_1^k)\hat{p}_k(T) \\ \mathbb{E}(R_2) &= \textstyle\sum_k h_2(z_2^k)\hat{p}_k(T) \\ &\dots \\ \mathbb{E}(R_m) &= \textstyle\sum_k h_m(z_m^k)\hat{p}_k(T)) \end{aligned} \tag{5}$$

where $z_1^k, z_2^k, \dots z_m^k$ are the values of the performance metrics for the non-failed system configurations $k \in \overline{FC}$, with $\overline{FC}$ being the set of system non-failed configurations, and $\hat{p}_k(T)$ the conditional probability that the system has reached the non-failed configuration k given that the system is in any of its possible non-failed configurations $\overline{FC}$:

$$\hat{p}_k(T) = \frac{p_k(T)}{\Sigma_{k \in \overline{FC}} p_k(T)}. \tag{6}$$

III. MATLAB/SIMULINK Evaluation Tool

In order to illustrate the proposed methodology, we developed a MATLAB/SIMULINK® based tool. This tool implements all the elements of the methodology described in Section II, enabling the automatic evaluation of any system. The SIMULINK® environment enables the construction of the system behavioral model and the definition of performance metrics as well as requirements for them. A collection of scripts coded in MATLAB® carries out the performance evaluation and the calculation of probabilistic measures of performance.

A. System Behavioral Model Definition

The system behavioral model is defined in the SIMULINK® environment. In section II, it was mentioned that the system configurations dynamic models, (1), is one of the elements of the methodology. Theoretically it is necessary to have all the system configurations beforehand to carry out the analysis of a system. In practice, the system nominal configuration ($i = 1$ in equation 1, no failures) is modeled in SIMULINK® by interconnecting the nominal models of each component, and afterwards each component model is augmented with an artifact to model the component failure behavior. The realization of the remaining system configurations is carried out through simulation by injecting different sequences of component failure modes ($i = 2 \dots N$ in (1)).

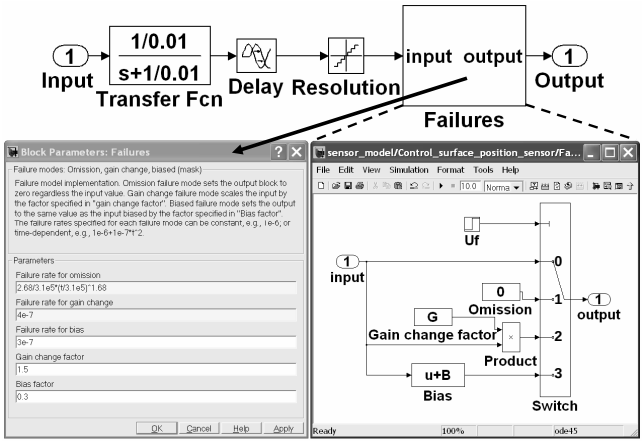

Figure 1. SIMULINK® Control Surface Position Sensor Model.

To illustrate how a system model is defined, we will show how to define one of the components within a model. The complete system model will result after connecting all the component models. Figure 1 shows a SIMULINK® model for one of the control surface position sensors of the case-study presented in Section IV. The mathematical model from which this example was developed is shown in Appendix VI. On the upper part of Figure 1, the elements labeled *Transfer Fcn*, *Delay*, and *Resolution* are common SIMULINK® blocks that allow the sensor functional properties to be modeled.

The block on the upper-right of Figure 1, named *Failures*, is just an artifact to model the component failure behavior. The content of the *Failures* block is displayed on the right bottom of Figure 1. The

failure behavior of the sensor depends on the *Switch* control input U_f, which is controlled by the simulation environment created in MATLAB®. For $U_f = 0$, the *Switch* block input 0 is passed to the output, meaning the sensor is failure-free. When the *Switch* control input U_f is set to $1, 2,$ or 3, the inputs $1, 2, 3$ of the *Switch* block are passed to the output, modeling, respectively, Omission, Gain-Change and Bias failure modes.

The bottom-left of Figure1 shows the GUI through which the user interfaces with the *Failures* block, allowing the definition of failure rates for each failure mode –in this case, Omission, Gain-change, and Bias. It is possible to define constant failure rates and time-dependent failure rates if component wear-out mechanisms are to be modeled. The *Failures* block GUI allows other parameters of the component failure model to be defined –in this case, the Gain-change factor and the Bias factor. The component can only go from the non-failed status to any of the failed statuses defined by the failure modes, but once it is in a failed status, it remains there –it cannot fail in a different mode. However, it is possible to refine the tool and allow transitions, not only from the non-failed status to any failure status, but between any failed status.

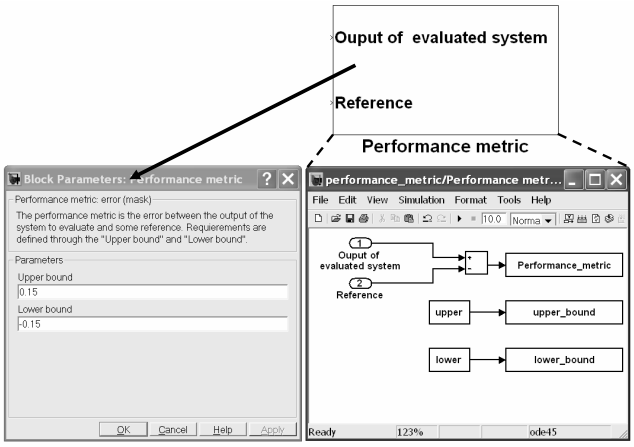

Figure 2. SIMULINK® performance metric definition.

B. Performance Metrics Definition

The performance metrics, as well as their requirements, are defined in SIMULINK®. Figure 2 illustrates how to define a performance metric. The content of the *Performance metric* block is displayed on the right bottom of Figure 2, which defines the performance metric as the difference between the output of the system under evaluation and a reference. The bottom-left of the figure shows the GUI through which the user interfaces with the *Performance metric* block, allowing the definition of the upper and lower bounds.

C. Performance Evaluation and Probabilistic Measures of Performance

For each possible system configuration, the system is simulated in order to check whether or not the performance requirements are met. At the same time the performance of the system is evaluated for each sequence of failures, the state-transition matrix associated with (2) is built.

The steps followed by the tool can be summarized as:

1) Each possible single failure is injected in the SIMULINK® system behavioral model. Therefore n_1 new system configurations are realized, where n_1 is the sum of failure modes of all the system components.
2) For each system configuration realized in step 1 a new entry is added to the state-transition matrix associated with (2).
3) The behavior of the configurations realized in step 1 is simulated for a period of time, and based on the results of the simulation, the system is declared to be in a failed configuration or in a non-failed configuration.
4) From the component failures resulting in non-failed configurations in step 3, and the remaining non-failed components, sequences of two-failed elements are generated and injected in the SIMULINK® system behavioral model. There results n_2 new system configurations, where n_2 is the number of sequences with two different failed components (in a particular failure mode).
5) For each new system configuration realized in step 4 a new entry is added to the state-transition matrix associated with (2).
6) The behavior of the configurations realized in step 5 is simulated for a period of time, and based on the results of the simulation, the system is declared in a failed configuration or in a non-failed configuration.
7) From the sequences of failures resulting in non-failed configurations in step 6 and the remaining non-failed components, new sequences of three failed elements are generated and injected in the SIMULINK® system behavioral model.

8) This process lasts until all the sequences of component failures result in a failed system configuration or the analysis is truncated when the sequences of failed components have a predetermined size, i.e., number of failed elements within the sequence.

For a given sequence of failures s_k of size k (k components are failed) the system behavior is simulated for a period of time t_c (*configuration evaluation time*) and for a given set of system control inputs $u(t)$. At the the beginning of the simulation period, the system state variables $x(t)$ are set to the final values of the simulation period for the sequence of failures s_{k-1} that gave birth to the sequence s_k by introducing an additional yet-to-fail k^{th} component. After the simulation is running, the failure of component k is introduced at t_f (*failure injection time*) with $t_f < t_c$. The values of t_f and t_c are chosen such that the system reaches a new steady-state in $t_c - t_f$ if the new configuration is stable, or the system behavior rapidly diverts from the expected nominal behavior if the configuration is unstable. If the system configuration is unstable, or it is stable but the performance metrics do not meet requirements, the configuration is declared as failed. If the configuration is stable and meets performance requirements, then it is declared as a non-failed configuration.

Once all the system configurations are evaluated, the probabilities $p_i(T)$ of reaching, at time T, each system configuration, i, from the nominal configuration at time $T = 0$ can be computed. A reward model, (4), is built with the values of the performance metrics for each system configuration. Finally, probabilistic measures of performance are computed using (5). The outputs for each system configuration are:

- The sequence of component failures leading to that system configuration.
- Probability of being in that system configuration at the global evaluation time T.
- The performance metrics values in that system configuration.
- A tag to indicate whether the configuration is failed or not.

Additionally, System reliability and unreliability values are collected, as well as the probabilistic measures of performance associated with each performance metric. The tool also allows each particular system configuration to simulated individually.

IV. Case-Study

In this section, we present a case-study of a fault-tolerant architecture for a fighter aircraft lateral-directional flight control system [3]. For clarity, the system architecture under study is described in detail here, but the behavioral model equations for each component within the system are described in several appendices at the end of the paper.

The authors acknowledge the fact that the redundancy and the failure detection, isolation, and reconfiguration (FDIR) elements introduced in the system under study are, by no means, complete. However, one of the objects of the methodology presented in this paper is to uncover weak design points, e.g., not enough component redundancy, or insufficient FDIR to achieve fault-tolerance. Therefore, we use this architecture to show how the methodology presented in this paper can help guiding the redundancy allocation and the FDIR design.

A. System Behavioral Model Definition

The behavioral model of the architecture was defined using SIMULINK®, Figure 8 (last page of the paper). The proposed architecture is composed of two redundant primary flight computers (PFC) (Appendix I) that receive information about aircraft attitude from three redundant inertial measurement units (IMU) (Appendix II) and from triple redundant control surface (rudder, right aileron and left aileron) position sensors (Appendix VI) and reconfiguration signals from the other PFC. Both PFCs have a voting algorithm (Appendix I) implemented to compute the actual aircraft attitude from the redundant IMU measurements. Both PFCs have also implemented control laws (Appendix I) that, based on the IMU measurements and the pilot inputs, compute the appropriate commands for the control surface actuation subsystems (Appendix III). Each control surface (Appendix V) is actuated by two redundant actuation subsystems commanded independently from each PFC, which are linked to the corresponding control surface by a mechanical combiner (Appendix IV). Each actuation subsystem is composed of a current-controlled electric motor. Therefore, there are three actuation subsystems per PFC, commanding both left and right ailerons, and the rudder.

As shown above, every component, except the control surfaces, is duplicated or triplicated in order to achieve fault-tolerance. However, though redundancy is an important element of any fault-tolerant system,

there is another equally important element – failure detection, isolation and reconfiguration (FDIR). In this case-study, some FDIR elements have been introduced to illustrate how the system will recover from some failures, and will not recover from others despite the presence of redundancy. The PFCs have a failure detection circuit able to detect a failure-by-omission in the computer. When this happens, the failure-detection circuit shuts down the main computer processor and sends a reconfiguration signal to the other computer to increase the gain of the control surface controllers to compensate for the fact that now only the control surface actuation subsystems commanded by the remaining computer are driving the control surfaces.

Table 1 collects information corresponding to the failure models of the different components. The actual failure models are part of the component behavioral models described in Appendices I-VI. The possible failure modes of each component are listed in column 2 of Table 1, while column 3 is an explanation of the effect of each failure mode on the component behavior. U_f in column 4 is the variable that assigns the corresponding failure mode to the component behavioral model equations (see Appendices I-VI). The last column of Table 1 collects the failure rates λ associated with each failure mode. These failure rates will allow the construction of the state-transition matrix associated with (2).

TABLE 1

COMPONENT FAILURE MODEL PARAMETERS.

Component	Failure modes	Description	U_f	$\lambda(/h)$
Primary_flight_computer_i(i=1,2)	Omission	Output set to zero	1	$2 \cdot 10^{-7}$
	Random	Random output between -5 and 5	2	10^{-7}
	Stuck	Output stuck at last correct value	3	10^{-7}
	Delayed	Output delayed 0.2 s	4	10^{-7}
PFC_i_self_check_circuit(i=1,2)	Omission	Output set to zero regardless the input	1	10^{-8}
	Commission	Output set to one regardless the input	2	10^{-8}
Left_aileron_actuation_subsystem_i(i=1,2) Right_aileron_actuation_subsystem_i(i=1,2) Rudder_actuation_subsystem_i(i=1,2)	Omission	Output set to zero	1	10^{-6}
	Stuck	Output stuck at last correct value	2	10^{-6}
Left_aileron, Right_aileron, Rudder	Omission	Output set to zero	1	10^{-8}
	Trailing	Output commanded by the aircraft dynamics	2	10^{-8}
Inertial_measurement_unit_i(i=1,2,3)	Omission	Output set to zero	1	$4 \cdot 10^{-7}$
	Gain change	Output scaled by a factor of 1.5	2	$3 \cdot 10^{-7}$
	Biased	Output biased by a factor of 0.3	3	$3 \cdot 10^{-7}$
Left_aileron_angle_sensor_i(i=1,2,3) Right_aileron_angle_sensor_i(i=1,2,3) Rudder_angle_sensor_i(i=1,2,3)	Omission	Output set to zero	1	$4 \cdot 10^{-7}$
	Gain change	Output scaled by a factor of 1.5	2	$3 \cdot 10^{-7}$
	Biased	Output biased by a factor of 0.3	3	$3 \cdot 10^{-7}$

To complete the system behavioral model, a linear lateral-directional aircraft dynamic model [12], [13] interacting with the avionics architecture model described above is included. The state variables of this model are the sideslip β, the body axis roll rate p_b, the body axis yaw rate r_b, and the body axis roll angle ϕ. The control surface commands are both left and right aileron angles δ_a^l and δ_a^r, and the rudder angle δ_r. The complete state-space representation of the aircraft lateral-directional dynamics is shown in Appendix VII.

B. Performance Metrics Definition

The system nominal configuration (no failures) will be used to define the performance metrics. In the nominal configuration, the aircraft state variables are denoted by the super index ^{0}, i.e., the sideslip is denoted by β^0, the body axis roll angle by p_b^0, the body axis yaw rate by r_b^0, and the body axis roll angle by ϕ^0. The performance metrics are defined as maximum difference (positive or negative) between the actual state aircraft state variables and the nominal configuration ones and are denoted by

$$\begin{aligned} Z_\beta &= \beta(t) - \beta^0(t) \\ Z_{p_b} &= p_b(t) - p_b^0(t) \\ Z_{r_b} &= r_b(t) - r_b^0(t) \\ Z_\phi &= \phi(t) - \phi^0(t). \end{aligned} \tag{7}$$

The requirements of these performance metrics are defined as

$$\begin{aligned} L_\beta < Z_\beta < U_\beta \\ L_{p_b} < Z_{p_b} < U_{p_b} \\ L_{r_b} < Z_{r_b} < U_{r_b} \\ L_\phi < Z_\phi < U_\phi \end{aligned} \tag{8}$$

where U_β = 0.15rad, L_β = -0.15rad, U_{p_b} = 0.15rad, L_{p_b} = -0.15rad/s, U_{r_b} = 0.15rad, L_{r_b} = -0.15rad/s, and U_ϕ = 0.15rad, L_ϕ = -0.15rad.

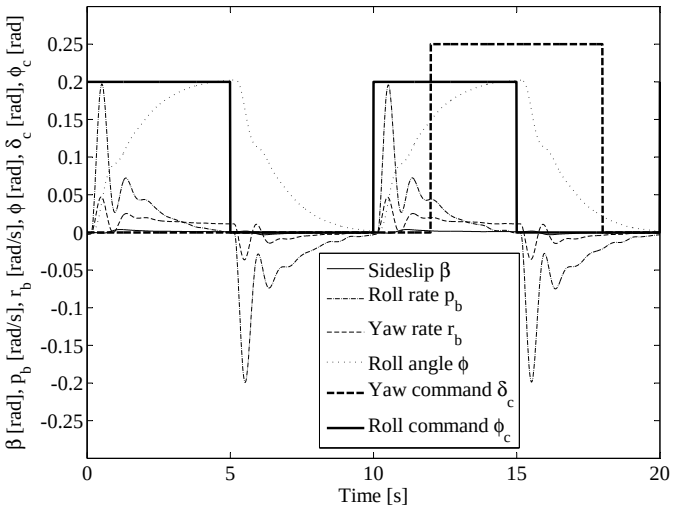

Figure 3. **Sideslip β, roll rate p_b, yaw rate r_b, and roll angle ϕ, versus time for a $0.2rad$, $0.1Hz$ square wave in the roll command ϕ_c, and a $5s$ width pulse in the yaw command δ_c for the system in its nominal configuration.**

C. System Evaluation and Results Analysis

The system evaluation was carried under specific conditions. The aircraft is considered to be in a cruising phase with forward velocity $V = 178$m/s, pitch angle $\alpha_0 = 0.216$rad and and a cruising altitude of $10,668$m. Under these conditions, the aircraft time constants dictate that a configuration evaluation time $t_c = 20$s, and a failure injection time $t_f = 4$s are appropriate. Therefore, in the remaining 16s of simulation, the system must reach a new steady-state if the new configuration is stable, or the system behavior must rapidly divert from the expected nominal behavior. The system control inputs considered for evaluating each configuration, as well as the aircraft dynamic response for the nominal configurations are displayed in Figure 3. The maintenance period of the aircraft was considered as the system global evaluation time T, taking a value of 500h.

Table 2 shows the probability of system failure (unreliability) at the end of the maintenance period for different levels of truncation. It can be see that it is enough to evaluate up to system configurations with three components failed. Truncating after three component failure events yields a system unreliability of $3.20 \cdot 10^{-4}$ and a truncation error of $2.39 \cdot 10^{-7}$. There is a trade-off between achieving a higher accuracy in estimating reliability and computational time for performing the evaluation. By truncating the evaluation at the second level, only 67 system configurations are evaluated, and the evaluation takes less than 3 minutes. If the truncation is done at the third level, 3675 possible configurations are analyzed and the evaluation takes 3 hours and 15 minutes. The computation was carried out on a machine with a 2.1 GHz Pentium® M processor, and 1.5Gb of RAM.

TABLE 2

SYSTEM UNRELIABILITY FOR DIFFERENT LEVELS OF TRUNCATION AND AN EVALUATION TIME OF $500h$.

Trunc. level	Unreliability Lower bound	Unreliability Upper bound	Truncation Error	# of configurations
2	$3.13 \cdot 10^{-4}$	$3.85 \cdot 10^{-4}$	$7.17 \cdot 10^{-5}$	67
3	$3.20 \cdot 10^{-4}$	$3.20 \cdot 10^{-4}$	$2.39 \cdot 10^{-7}$	3675

Table 3 displays the values of the probabilistic measures of performance corresponding to different truncation levels. In this case, the probabilistic measures of performance are the expected values of the system performance metrics given the system is in a non-failed configuration. Therefore, these measures give an idea of the system average deviation from nominal behavior. It is important to note that the nominal configuration does not affect these results since Z_β, Z_{p_b}, Z_{r_b}, and Z_ϕ are equal to zero in the nominal configuration. This is relevant, otherwise the result would be heavily biased since the probability of being in the nominal configuration is $p_1(500) = 0.988$, whereas the probability of being in any configuration with one component failed is on the order of 10^{-4}. For this same reason, the results $\mathbb{E}[Z_\beta]$, $\mathbb{E}[Z_{p_b}]$, $\mathbb{E}[Z_{r_b}]$, and $\mathbb{E}[Z_\phi]$ do not significantly change by truncating the evaluation at the second level of failure instead of at the third level. Therefore, it is necessary to investigate whether it is more appropriate to compute these expectations at each level of failure.

TABLE 3

PROBABILISTIC MEASURES OF PERFORMANCE: EXPECTED VALUES OF THE PERFORMANCE METRICS FOR DIFFERENT LEVELS OF TRUNCATION.

Trunc. level	$\mathbb{E}[Z_\beta]$	$\mathbb{E}[Z_{p_b}]$	$\mathbb{E}[Z_{r_b}]$	$\mathbb{E}[Z_\phi]$
2	$6.30 \cdot 10^{-6}$	$3.24 \cdot 10^{-4}$	$8.52 \cdot 10^{-5}$	$6.00 \cdot 10^{-5}$
3	$6.39 \cdot 10^{-6}$	$3.27 \cdot 10^{-4}$	$8.61 \cdot 10^{-5}$	$6.08 \cdot 10^{-5}$

Figure 4 shows the system performance metrics Z_β, Z_{p_b}, Z_{r_b}, and Z_ϕ for a single failure in the left aileron, in which it fails by getting stuck at the position it was in when the failure occurred. Although the system performance is degraded, the performance

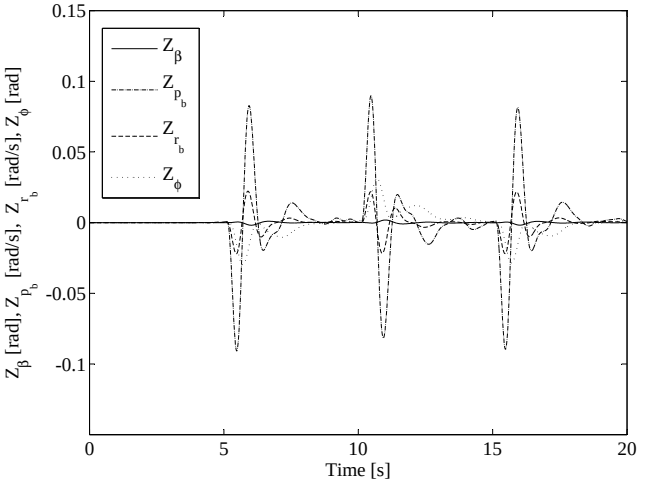

Figure 4. **Performance metrics Z_β, Z_{p_b}, Z_{r_b}, and Z_ϕ for a stuck failure mode in the left or right aileron, versus time for a 0.2rad, 0.1Hz square wave in the roll command ϕ_c, and a 5s width pulse in the yaw command δ_c.**

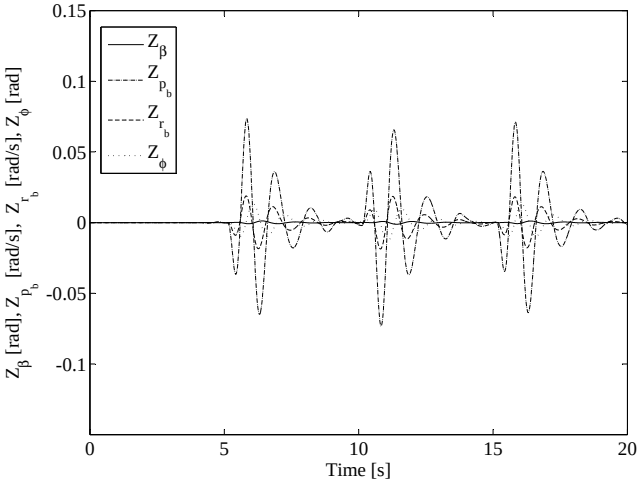

Figure 6. **Performance metrics Z_β, Z_{p_b}, Z_{r_b}, and Z_ϕ for a output omission failure mode in one of the left or right aileron actuation subsystems, versus time for a 0.2rad, 0.1Hz square wave in the roll command ϕ_c, and a 5s width pulse in the yaw command δ_c.**

metrics remain within their requirements (0.15rad or -0.15rad). In this case, there are not many things that can be done to improve the system performance since the aileron is non-redundant, and when it fails, it affects the aircraft aerodynamics. It might be possible to modify the control laws to account for a failure of this type.

Figure 5 shows the performance metrics for another failure of the left aileron. The failure mode is such that the aileron is now commanded by the aircraft

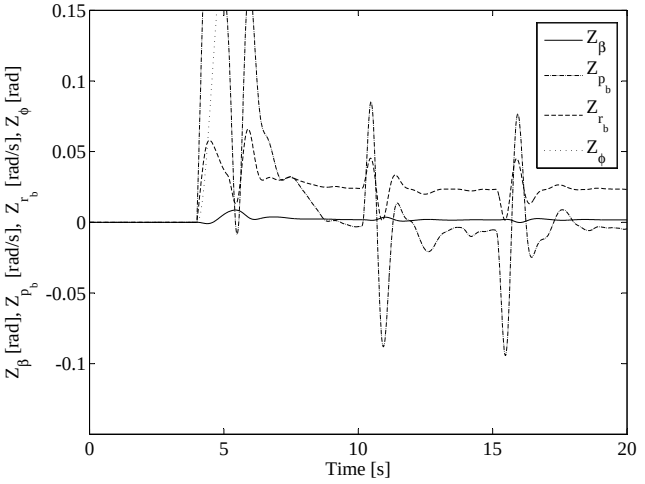

Figure 5. **Performance metrics Z_β, Z_{p_b}, Z_{r_b}, and Z_ϕ for a trailing failure mode in the left or right aileron, versus time for a 0.2rad, 0.1Hz square wave in the roll command ϕ_c, and a 5s width pulse in the yaw command δ_c.**

dynamics, i.e., the aileron trails. In this case, the failure is catastrophic. It can be seen that 4s after the failure occurs, Z_{r_b} rapidly increases. This means that the aircraft is rolling without any control. It can be seen that Z_ϕ increases as well, which is a natural result since ϕ is the aircraft rolling angle.

Figure 6 corresponds to a failure by omission of one of the left aileron actuation subsystems. This results in degraded system performance. Nevertheless the configuration is stable and meets the performance requirements. Therefore this configuration is considered as non-failed. It would be possible to have perfect system performance when failures occur within the actuation subsystems by implementing FDIR mechanisms to account for these failures.

With FDIR implemented when any of the primary flight computers fails by output omission, the performance metrics remain smaller than $3 \cdot 10^{-3}$ in absolute value. Therefore the system behavior is almost unaffected. This is expected since computer output omission failures can be detected and isolated with the self-detection circuit built in each computer, and the control laws in the remaining computer reconfigured to account for this type of failure.

Figure 7 shows the results for another type of failure within the computer. In this case, it is a failure that causes the computer to output a random command for the control surface actuation subsystems. Unlike the –failure-by-output-omission, this failure is not

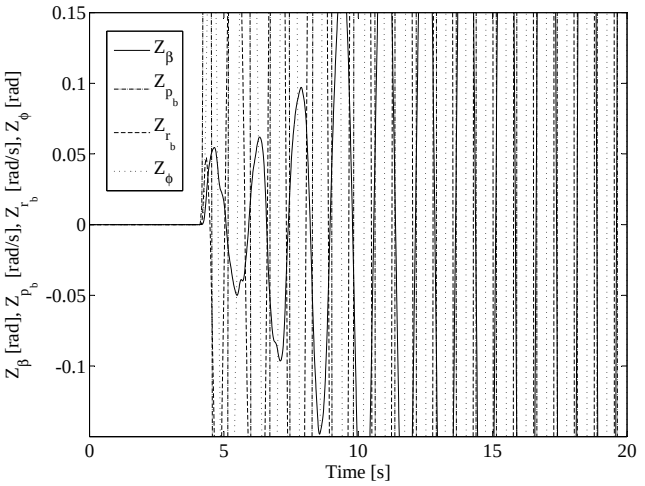

Figure 7. Performance metrics Z_β, Z_{p_b}, Z_{r_b}, and Z_ϕ for a random output failure mode in one of primary flight computers, versus time for a 0.2rad, 0.1Hz square wave in the roll command ϕ_c, and a 5s width pulse in the yaw command δ_c.

survivable, the system becomes unstable resulting in a failed configuration. A way to remove this failed configuration could be by improving the self-detection circuit of each computer to allow the detection and isolation of this type of failure within the computer.

V. Conclusions and Future Work

In this paper, we demonstrate that model-based performance evaluation of fault-tolerant systems is a powerful methodology for the design phase of fault-tolerant systems. By including other important system performance metrics and relying on behavioral and objective models of the system, the system performance and reliability is evaluated in a more thorough fashion than more traditional techniques based on system functional models. We prove the feasibility of the methodology by showing a tool developed using MATLAB/SIMULINK® that performs the system evaluation automatically.

There are still several issues that need to be investigated. As mentioned before, it is not clear if the probabilistic measures of performance must be calculated by aggregating configurations with the same number of components failed or as an aggregate of all non-failed configurations across all the levels of failure. So far, the methodology only addresses how to evaluate a given system architecture, but it does not address in a structured way how to improve the architecture. It is true that the methodology uncovers weak points in the design and might help improve certain parts to eliminate single points of failure or improve the performance of certain non-failed configurations. However, there is nothing in the methodology that points out the system unreliability drivers, i.e., which component failures are driving the system reliability or the performance of the non-failed configurations. Therefore it is necessary to investigate how to incorporate new elements to the methodology to help improve the design in an structured way.

Acknowledgment

Some of the ideas presented in Section III were developed in collaboration with Benjamin Sewell and Dr. Peter Miller from Ricardo UK. The authors would like to thank Dr. Philip S. Babcock at the Charles Stark Draper Laboratory for his invaluable ideas and fruitful discussions, and for his constant support of this research. Piero Miotto at the Charles Stark Draper Laboratory provided the IMU model and fruitful discussions. Thanks also to Thomas T. Myers at Systems Technology, Inc. for providing technical information regarding the linear lateral-directional aircraft dynamics model.

References

[1] J. Doyle, B. Francis, and A. Tannenbaum, *Feedback Control Theory*. New York, NY: Macmillan, 1992.

[2] K. Zhou and J. Doyle, *Essentials of Robust Control*. Upper Saddle River, NJ: Prencice Hall, 1998.

[3] D. McRuer, T. Myers, and P. Thompson, "Literal singular-value-based flight control system design techniques," *AIAA Journal of Guidance*, vol. 12, no. 6, pp. 913–919, 1989.

[4] A. Høyland and M. Rausand, *System Reliability Theory*. New York, NY: John Wiley and Sons, 1994.

[5] Y. Papadopoulos, J. McDermid, R. Sasse, and G. Heiner, "Analysis and synthesis of the behavior of complex programmable electronic systems in conditions of failure," *Journal of Reliability Engineering and System Safety*, vol. 71, no. 3, pp. 229–247, Mar. 2001.

[6] Y. Papadopoulos, D. Parker, and C. Grante, "Model-based automated synthesis of fault trees from matlab-simulink models," in *Proceedings of the International Conference on Dependable Systems and Networks*, Gothenburg, Sweden, 2001, pp. 77–82.

[7] N. Leveson, *Safeware: System Safety and Computers*. Boston, MA: Addison-Wesley Publishing Company, 1995.

[8] A. Amendola, "Event sequences and consequence spectrum: A methodology for probabilistic transient analysis," *Nuclear Science An Engineering*, vol. 77, no. 3, pp. 297–315, 1981.

[9] T. Aldemir, "Computer-assisted markov failure modeling of process control systems," *IEEE Transactions on Reliability*, vol. R-36, no. 1, pp. 133–144, Apr. 1987.

[10] J. Devooght and C. Smidts, "Probabilistic reactor dynamics-I: The theory of continuos event trees," *Nuclear Science and Engineering*, vol. 111.

[11] ——, "Probabilistic dynamics as a tool for dynamic psa," *Reliability Engineering and System Safety*, vol. 52, no. 3, pp. 185–196, June 1996.
[12] D. McRuer, I. Ashekenas, and D. Graham, *Aircraft Dynamics and Automatic Control*. Princeton, NJ: Princeton University Press, 1973.
[13] J. Roskan, *Flight Dynamics of Rigid and Elastic Airplanes*. Lawrence, KS: The University of Kansas, 1972.
[14] D. McRuer and T. Myers, "Advanced piloted aircraft flight control system design methodology volume I: Knowledge base," NASA, Langley, VA, Contractor Report 181726, 1988.

EMAIL

Alejandro D. Domínguez-García: aledan@MIT.EDU

APPENDIX I
PRIMARY FLIGHT COMPUTER

A. Voter

$$\begin{aligned}\epsilon_1 &= |u_1 - u_2| \\ \epsilon_2 &= |u_1 - u_3| \\ \epsilon_3 &= |u_2 - u_3|\end{aligned}$$

$$\tilde{u} = \begin{cases} \sum_{i=1}^{3} u_i - \max_i\{u_i\} - \min_i\{u_i\}, & \text{if } \exists\ i,j\ /\ \epsilon_i + \epsilon_j < 2\epsilon \\ \frac{u_i + u_j}{2}, & \text{if } \exists\ i\ /\ \epsilon_i < \epsilon \\ NIL, & otherwise \end{cases} \quad (9)$$

where $\epsilon = 0.1$rad.

B. Control Laws

1) Roll Control:

$$\begin{aligned}R_r(s) &= K_{r_1}\phi_c(s) + K_{r_2}R_b(s) + K_{r_3}\tfrac{s+z_r}{s+p_r}P_b(s) \\ \delta_{a_r^*}^{l(r)}(s) &= (P_r + \tfrac{I_r}{s} + D_r s)(R_r(s) \pm K_r\delta_a^{l(r)}(s))\end{aligned}$$

$$\delta_{a_r}^{l(r)}(t) = \begin{cases} \delta_{a_r^*}^{l(r)}(t), & \text{for } r = 0 \\ 2\delta_{a_r^*}^{l(r)}(t), & \text{for } r = 1 \end{cases}$$

$$\hat{\delta}_{a_r}^{l(r)}(t) = \begin{cases} \delta_{a_r}^{l(r)}(t), & \text{for } U_f = 0 \\ 0, & \text{for } U_f = 1 \\ rand(l,u), & \text{for } U_f = 2 \\ \delta_{a_r}^{l(r)}(\tau), & \text{for } U_f = 3 \\ \sigma_{\tau_l}(\delta_{a_r}^{l(r)}(t)), & \text{for } U_f = 4 \end{cases} \quad (10)$$

where $K_{r_1} = 0.66$, $K_{r_2} = -0.145$s, $K_{r_3} = 2.16$s, $z_r = 11.1\text{s}^{-1}$, $p_r = 25\text{s}^{-1}$, $P_r = 0.45$A, $I_r = 6$A/s, $D_r = 0.01$As, and $K_r = -1.33$ were taken from [3]; and $l = -5$A, $u = 5$A $\tau_l = 0.2$s, and τ is the time at which the computer gets stuck.

2) Yaw Control:

$$\begin{aligned}R_y(s) &= K_{y_1}\delta_c(s) + \tfrac{s^2+z_{y_1}s+z_{y_2}}{s(s+p_y)}(K_{y_2}R_b(s) + K_{y_3}P_b(s)) \\ \delta_{r_r^*}(s) &= (P_y + \tfrac{I_y}{s} + D_y s)(R_y(s) + K_y\delta_r(s))\end{aligned}$$

$$\delta_{r_r}(t) = \begin{cases} \delta_{r_r^*}(t), & \text{for } r = 0 \\ 2\delta_{r_r^*}(t), & \text{for } r = 1 \end{cases}$$

$$\hat{\delta}_{r_r}(t) = \begin{cases} \delta_{r_r}(t), & \text{for } U_f = 0 \\ 0, & \text{for } U_f = 1 \\ rand(l,u), & \text{for } U_f = 2 \\ \delta_{r_r}(\tau), & \text{for } U_f = 3 \\ \sigma_{\tau_l}(\delta_{r_r}(t)), & \text{for } U_f = 4 \end{cases} \quad (11)$$

where $K_{y_1} = 0.001$, $K_{y_2} = -0.7816$s, $K_{y_3} = 0.172$s, $z_{y_1} = -0.00125\text{s}^{-2}$, $z_{y_2} = -0.001875\text{s}^{-1}$, $p_y = 1.5\text{s}^{-1}$, $P_y = 0.45$A, $I_y = 6$A/s, $D_y = 0.01$As, $K_y = -1.33$, were taken from [3]; and $l = -5$A, $u = 5$A $\tau_l = 0.2$s, and τ is the time at which the computer gets stuck.

APPENDIX II
INERTIAL MEASUREMENT UNIT

$$\begin{aligned}x_{imu}(t+\Delta t) &= x_{imu}(t) + randn(0, \sigma_{ss}\sqrt{1 - e^{-2\Delta t/\tau}}) \\ x_{imu}(0) &= \sigma_{ss}\end{aligned}$$

$$\begin{aligned}y_{imu}(t+\Delta t) &= x_{imu}(t+\Delta t) + \\ &[M_1 + M_2 + I]\begin{bmatrix} p_b \\ q_b \\ r_b \end{bmatrix} + \begin{bmatrix} randn(\mu_b, \sigma_b) \\ randn(\mu_b, \sigma_b) \\ randn(\mu_b, \sigma_b) \end{bmatrix}\end{aligned}$$

$$M_1 = K_1\begin{bmatrix} randn(0,1) & 0 & 0 \\ 0 & randn(0,1) & 0 \\ 0 & 0 & randn(0,1) \end{bmatrix}$$

$$M_2 = K_2\begin{bmatrix} 0 & 0 & 0 \\ -randn(0,1) & 0 & 0 \\ randn(0,1) & 0 & -randn(0,1) \end{bmatrix}$$

$$\hat{y}_{imu} = \begin{cases} y_{imu}, & \text{for } U_f = 0 \\ 0, & \text{for } U_f = 1 \\ Gy_{imu}, & \text{for } U_f = 2 \\ y_{imu} + B, & \text{for } U_f = 3 \end{cases} \quad (12)$$

where $\sigma_{ss} = 3.15 \cdot 10^{-6}$, $\Delta t = 0.01$s, $\tau = 100$s, $\mu_b = 4.85 \cdot 10^{-6} rand(0,1)$, $\sigma_b = 2.09 \cdot 10^{-5}$, $K_1 = 10^{-4}$, $K_2 = 9.7 \cdot 10^{-5}$, $G = 1.5$, and $B = 0.3$deg.

APPENDIX III
ACTUATION SUBSYSTEM

3) Ailerons:

$$\begin{aligned}G(s) &= k_1\tfrac{(s+\frac{1}{\tau_c})(s+\frac{1}{\tau_m})}{(s+\frac{1}{\tau_1})(s+\frac{1}{\tau_2})(s+\frac{1}{\tau_3})} \\ T_{a_{1(2)}}^{l(r)}(s) &= k_2\tfrac{G(s)}{1+G(s)}\hat{\delta}_{a_r}^{l(r)}(s)\end{aligned}$$

$$\hat{T}_{a_{1(2)}}^{l(r)}(t) = \begin{cases} T_{a_{1(2)}}^{l(r)}(t), & \text{for } U_f = 0 \\ 0, & \text{for } U_f = 1 \\ T_{a_{1(2)}}^{l(r)}(\tau), & \text{for } U_f = 2 \end{cases} \quad (13)$$

where $k_1 = 5000\text{s}^2$, $k_2 = 50$, $\tau_c = 10\text{s}$, $\tau_m = 10\text{s}$, $\tau_1 = 4\text{s}$, $\tau_2 = 10\text{s}$, $\tau_3 = 100\text{s}$, and τ is the time at which the actuation subsystem gets stuck.

4) Rudder:

$$G(s) = k_1 \frac{(s+\frac{1}{\tau_c})(s+\frac{1}{\tau_m})}{(s+\frac{1}{\tau_1})(s+\frac{1}{\tau_2})(s+\frac{1}{\tau_3})}$$
$$T_{r_{1(2)}}(s) = k_2 \frac{G(s)}{1+G(s)} \hat{\delta}_r(s)$$

$$\hat{T}_{r_{1(2)}}(t) = \begin{cases} T_{r_{1(2)}}, & \text{for } U_f = 0 \\ 0, & \text{for } U_f = 1 \\ T_{r_{1(2)}}(\tau), & \text{for } U_f = 2 \end{cases} \quad (14)$$

where $k_1 = 5000\text{s}^2$, $k_2 = 50$, $\tau_c = 10\text{s}$, $\tau_m = 10\text{s}$, $\tau_1 = 4\text{s}$, $\tau_2 = 10\text{s}$, $\tau_3 = 100\text{s}$, and τ is the time at which the actuation subsystem gets stuck.

APPENDIX IV
MECHANICAL COMBINER

A. Ailerons

$$T_a^{l,r} = G_a(\hat{T}_{a_1}^{l(r)} + \hat{T}_{a_2}^{l(r)}) \quad (15)$$

where $G_a = 0.5$.

B. Rudder

$$T_r = G_r(\hat{T}_{r_1} + \hat{T}_{r_2}) \quad (16)$$

where $G_r = 0.5$.

APPENDIX V
CONTROL SURFACES

A. Ailerons

$$\dot{x}_a^{l,r} = -\frac{1}{\tau_a} x_a^{l,r} + \frac{1}{\tau_a} T_a^{l,r}$$
$$\delta_a^{l(r)} = \sigma_{\tau_l}(x_a)$$

$$\hat{\delta}_a^{l(r)} = \begin{cases} \delta_a^{l(r)}, & \text{for } U_f = 0 \\ \delta_a(\tau), & \text{for } U_f = 1 \\ \alpha_0, & \text{for } U_f = 2 \end{cases} \quad (17)$$

where $\tau_a = 0.04\text{s}$ and $\tau_l = 0.11\text{s}$, $\alpha_0 = 0.216\text{rad}$, were taken from [3]. τ is the time at which the aileron gets stuck.

B. Rudder

$$\dot{x}_r = -\frac{1}{\tau_r} x_r + \frac{1}{\tau_r} T_r$$
$$\delta_r = \sigma_{\tau_l}(x_r)$$

$$\hat{\delta}_r = \begin{cases} \delta_r, & \text{for } U_f = 0 \\ \delta_r(\tau), & \text{for } U_f = 1 \\ \Psi, & \text{for } U_f = 2 \end{cases} \quad (18)$$

where $\tau_r = 0.05\text{s}$ and $\tau_l = 0.1\text{s}$ were taken from [3]. τ is the time at which the rudder gets stuck.

APPENDIX VI
CONTROL SURFACES POSITION SENSOR

A. Ailerons

$$\dot{x}_{s_a}^{l(r)} = -\frac{1}{\tau_s} x_{s_a}^{l(r)} + \frac{1}{\tau_s} \delta_a^{l(r)}$$
$$y_{s_a}^{l(r)} = \frac{\lceil \sigma_{\tau_l}(x_{s_a}^{l(r)}) \frac{1}{R} \rceil}{\frac{1}{R}}$$

$$\hat{y}_{s_a}^{l(r)} = \begin{cases} y_{s_a}^{l(r)}, & \text{for } U_f = 0 \\ 0, & \text{for } U_f = 1 \\ G y_{s_a}^{l(r)}, & \text{for } U_f = 2 \\ y_{s_a}^{l(r)} + B, & \text{for } U_f = 3 \end{cases} \quad (19)$$

where $\tau_s = 0.01\text{s}$, $\tau_l = 0.001\text{s}$, $R = 0.001$, $G = 1.5$, and $B = 0.3\text{deg}$.

B. Rudder

$$\dot{x}_{s_r} = -\frac{1}{\tau_s} x_{s_r} + \frac{1}{\tau_s} \delta_r$$
$$y_{s_r} = \frac{\lceil \sigma_{\tau_l}(x_{s_r}) \frac{1}{R} \rceil}{\frac{1}{R}}$$

$$\hat{y}_{s_r} = \begin{cases} y_{s_r}, & \text{for } U_f = 0 \\ 0, & \text{for } U_f = 1 \\ G y_{s_r}, & \text{for } U_f = 2 \\ y_{s_r} + B, & \text{for } U_f = 3 \end{cases} \quad (20)$$

where $\tau_s = 0.01\text{s}$, $\tau_l = 0.001\text{s}$, $R = 0.001$, $G = 1.5$, and $B = 0.3\text{deg}$.

APPENDIX VII
LINEAR LATERAL-DIRECTIONAL AIRCRAFT DYNAMICS

$$\begin{bmatrix} \dot{\beta} \\ \dot{p}_b \\ \dot{r}_b \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \frac{Y_\beta}{V} & \sin\alpha_0 & -\cos\alpha_0 & \frac{g\cos\alpha_0}{V} \\ L_\beta & L_p & L_r & 0 \\ N_\beta & N_p & N_r & 0 \\ 0 & 1 & \tan\alpha_0 & 0 \end{bmatrix} \begin{bmatrix} \beta \\ p_b \\ r_b \\ \phi \end{bmatrix} + \begin{bmatrix} 0 & 0 & Y_{\delta_r} \\ N_{\delta_a^l} & N_{\delta_a^r} & N_{\delta_r} \\ L_{\delta_a^l} & L_{\delta_a^r} & L_{\delta_r} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \delta_a^l \\ \delta_a^r \\ \delta_r \end{bmatrix} \quad (21)$$

where $Y_\beta = -50.69m/s^2$, $V = 178m/s$, $\alpha_0 = 0.216rad$, $g = 9.81m/s^2$, $L_\beta = -32.3s^{-2}$, $L_p = -0.374s^{-1}$ $L_r = 2.40s^{-1}$, $N_\beta = 1.06s^{-2}$, $N_p = -0.0406s^{-1}$, $N_r = -0.0809s^{-1}$, $Y_{\delta_r} = 0.0179s^{-1}$, $N_{\delta_a^l} = -3.175s^{-2}$, $N_{\delta_a^r} = 3.175s^{-2}$, $N_{\delta_r} = 6.66s^{-2}$, $L_{\delta_a^l} = -0.855s^{-2}$, $L_{\delta_a^r} = 0.855s^{-2}$, and $L_{\delta_r} = -1.18s^{-2}$, were taken from [14].

APPENDIX VIII
NOTATION

B :	Bias factor for the IMUs and the control surface position sensors
$h_j(\cdot)$:	Reward model function for performance metric j
$f_i(\cdot,\cdot)$:	State evolution function for system configuration i
G :	Gain change factor for the IMUs and the control surface position sensors
G_a :	Left and right mechanical combiner gain
G_r :	Rudder mechanical combiner gain
g :	Acceleration of gravity
$g_i(\cdot,\cdot)$:	Instantaneous output function for system configuration i
L_β :	Dimensional variation of rolling moment about X_s with β
L_p :	Dimensional variation of rolling moment about X_s with p
L_r :	Dimensional variation of rolling moment about X_s with r
$L_{\delta_a^l}$:	Dimensional variation of rolling moment about X_s with δ_a^l
$L_{\delta_a^r}$:	Dimensional variation of rolling moment about X_s with δ_a^r
L_{δ_r} :	Dimensional variation of rolling moment about X_s with δ_r
m :	Number of system performance metrics
N :	Number of system configurations
N_β :	Dimensional variation of yawing moment about Z_s with β
N_p :	Dimensional variation of yawing moment about Z_s with p
N_r :	Dimensional variation of yawing moment about Z_s with r
$N_{\delta_a^l}$:	Dimensional variation of yawing moment about Z_s with δ_a^l
$N_{\delta_a^r}$:	Dimensional variation of yawing moment about Z_s with δ_a^r
N_{δ_r} :	Dimensional variation of yawing moment about Z_s with δ_r
L_j, U_j :	Lower and upper bound requirements for performance metric j
$p_b, P_b(s)$:	Roll rate
$p_i(T)$:	Probability that the system configuration at time T is i given that at $T = 0$ is 1
r :	Primary Flight Computers reconfiguration signals
$r_b, R_b(s)$:	Yaw rate
r_j :	Reward model associated with performance metric j
R :	Control surface position sensors resolution
s :	Laplace transform variable
t :	Time
t_c :	Configuration evaluation time
t_f :	Failure injection time
T :	System global evaluation time
$T_a^{l(r)}$:	Left (right) aileron torque command
$T_{a_{1(2)}}^{l(r)}, \hat{T}_{a_{1(2)}}^{l(r)}$:	Left (right) aileron actuation subsystem torque output
T_r :	Rudder torque command
$T_{r_{1(2)}}, \hat{T}_{r_{1(2)}}$:	Rudder actuation subsystem torque output
u_1, u_2, u_3 :	Voter inputs
$\tilde{u}$:	Voter output
$u(t)$:	System input
U_f :	Component failure model switch control input
V :	Forward velocity
$x(t)$:	System state variables
$x_a^{l,r}$:	Left (right) aileron state variable
x_{imu} :	Inertial measurement unit state variables
x_r :	Rudder state variable
$x_{s_a}^{l(r)}$:	Left (right) aileron position sensor state variable
x_{s_r} :	Rudder position sensor state variable
X_s, Y_s, Z_s :	Aircraft stability axis
$y(t)$:	System output
$y_{imu}, \hat{y}_{imu}$:	Inertial measurement output
$y_{s_a}^{l(r)}, \hat{y}_{s_a}^{l(r)}$:	Left (right) aileron position sensor output
$y_{s_r}, \hat{y}_{s_r}$:	Rudder position sensor output
Y_β :	Dimensional variation of Y_s-force with β
Y_{δ_r} :	Dimensional variation of Y_s-force with δ_r
Z_j :	Performance metric j
z_j^i :	Value of performance metric j for system configuration i
α_0 :	Pitch angle
β :	Sideslip angle
δ_c :	Yaw command
$\delta_a^{l(r)}, \hat{\delta}_a^{l(r)}$:	Left (right) aileron angle
$\delta_r, \hat{\delta}_r$:	Rudder angle
$\delta_{a_r^*}^{l(r)}, \delta_{a_r}^{l(r)}, \hat{\delta}_{a_r}^{l(r)}$:	Roll control output
$\delta_{r_r^*}, \delta_{r_r}, \hat{\delta}_{r_r}$:	Yaw control output
ϵ :	Voter tolerance error
λ :	Failure rate
λ_{ik} :	Transition rate from system configuration i to system configuration k
λ_k :	Transition rate out of system configuration k
ϕ :	Roll angle
ϕ_c :	Roll command
$\sigma_{\tau_l}(\cdot)$:	τ_l-shift operator
Ψ :	Heading angle
τ :	Failure time
τ_a:	Inverse of aileron bandwidth
τ_r:	Inverse of rudder bandwidth
τ_s :	Inverse of left, right, and rudder position sensor bandwidth
$\mathbb{E}[\cdot]$:	Expectation
$\lceil\cdot\rceil$:	Ceiling function
$rand(l,u)$:	Uniformly distributed random generator between l and u
$randn(\mu,\sigma)$:	Gaussian distributed random generator with mean μ and standard deviation σ
$k_1, k_2, \tau_1, \tau_2, \tau_3,$ τ_c, τ_m :	Left, right, and rudder actuation subsystem parameters
$K_1, K_2, \Delta t, \tau, \mu_b,$ σ_{ss}, σ_b :	IMU parameters
$K_{r_1}, K_{r_2}, K_{r_3},$ $z_r, p_r, P_r, I_r,$ D_r, K_r :	Roll control law parameters
$K_{y_1}, K_{y_2}, K_{y_3},$ $z_{y_1}, p_{y_2}, p_y, P_y,$ I_y, D_y, K_y :	Yaw control law parameters

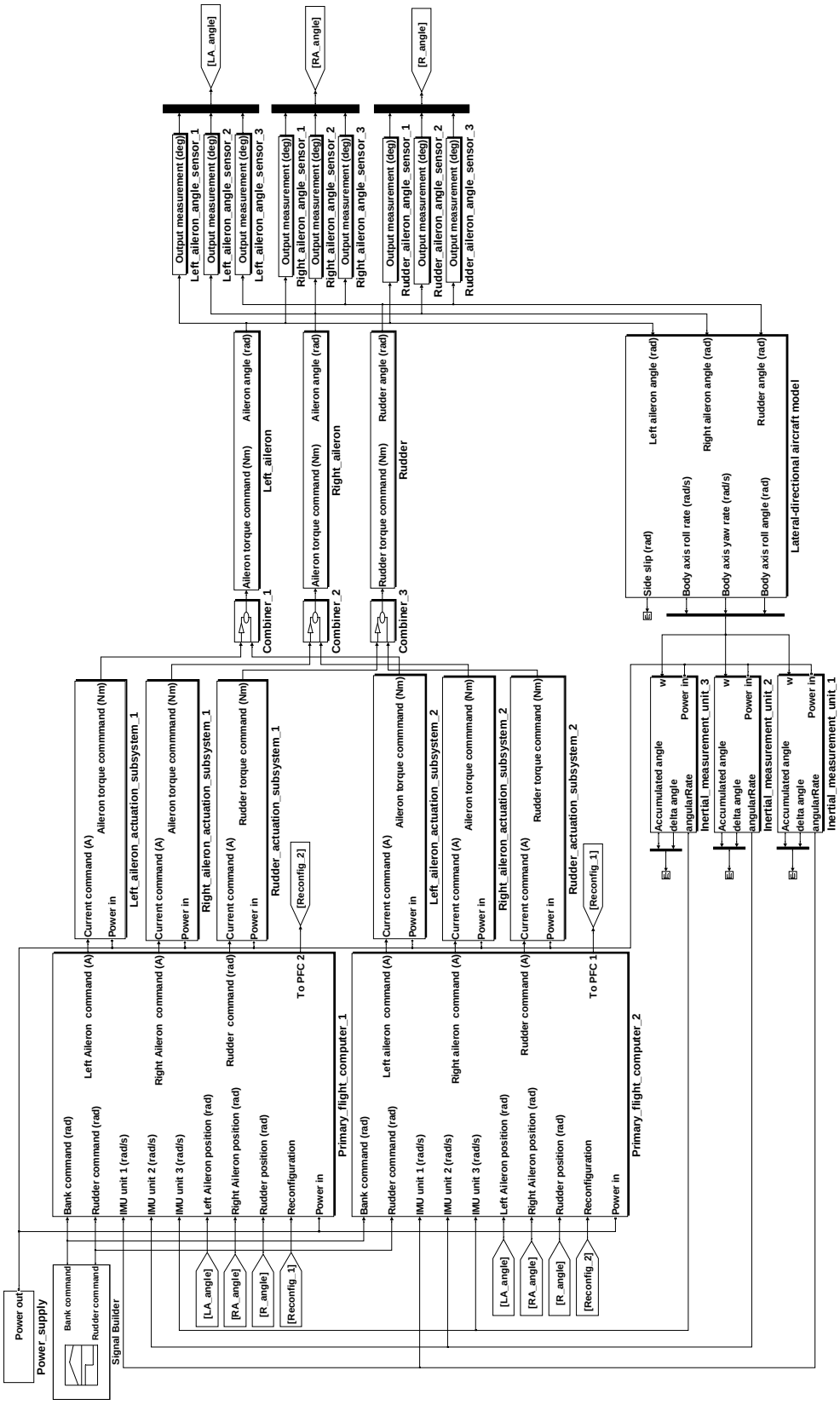

Figure 8. Lateral-Directional Flight Control System Fault-Tolerant Architecture.