# A simulation environment for design and testing of aircraft adaptive fault-tolerant control systems

*M.G. Perhinschi, M.R. Napolitano and G. Campa*
Department of Mechanical and Aerospace Engineering, West Virginia University, Morgantown, West Virginia, USA

**Abstract**
**Purpose** – The purpose of this paper is to present the development of a Matlab/Simulink-based simulation environment for the design and testing of indirect and direct adaptive flight control laws with fault tolerant capabilities to deal with the occurrence of actuator and sensor failures.
**Design/methodology/approach** – The simulation environment features a modular architecture and a detailed graphical user interface for simulation scenario set-up. Indirect adaptive flight control laws are implemented based on an optimal control design and frequency domain-based online parameter estimation. Direct adaptive flight control laws consist of non-linear dynamic inversion performed at a reference nominal flight condition augmented with artificial neural networks (NNs) to compensate for inversion errors and abnormal flight conditions following the occurrence of actuator or sensor failures. Failure detection, identification, and accommodation schemes relying on neural estimators are developed and implemented.
**Findings** – The simulation environment provides a valuable platform for the evaluation and validation of fault-tolerant flight control laws.
**Research limitations/implications** – The modularity of the simulation package allows rapid reconfiguration of control laws, aircraft model, and detection schemes. This flexibility allows the investigation of various design issues such as: the selection of control laws architecture (including the type of the neural augmentation), the tuning of NN parameters, the selection of parameter identification techniques, the effects of anti-control saturation techniques, the selection and the tuning of the control allocation scheme, as well as the selection and tuning of the failure detection and identification schemes.
**Originality/value** – The novelty of this research efforts resides in the development and the integration of a comprehensive simulation environment allowing a very detailed validation of a number of control laws for the purpose of verifying the performance of actuator and sensor failure detection, identification, and accommodation schemes.

**Keywords** Fault tolerance, Flight, Simulation, Flight control, Neural nets

**Paper type** Research paper

## 1. Introduction

Extensive research efforts have been conducted in recent years toward the design and the development of intelligent flight control laws able of handling failures and malfunctions of actuators and sensors. Within these efforts a variety of adaptive flight control methodologies have been developed (Steinberg, 1999). Adaptive flight control can be conceptually categorized as indirect, direct, and hybrid. Indirect approaches (Hageman *et al.*, 2003; Monaco *et al.*, 2004; Boskovic *et al.*, 2001) rely on real-time parameter identification (PID) for parameter adaptation followed by the reconfiguration of the control laws. Direct approaches (Calise and Sharma, 2000; Kaneshige *et al.*, 2000; Krishnakumar *et al.*, 2003; Williams-Hayes, 2005) use instead measurable states and tracking errors for adaptation and compensation purposes. Finally, the more recently introduced hybrid adaptive control methods (Nguyen *et al.*, 2006, 2007) combine elements of both indirect and direct methods.

The current issue and full text archive of this journal is available at
**www.emeraldinsight.com/1748-8842.htm**

Aircraft Engineering and Aerospace Technology: An International Journal
80/6 (2008) 620–632
© Emerald Group Publishing Limited [ISSN 1748-8842]
[DOI 10.1108/00022660810911563]

The challenging task of developing a comprehensive, integrated solution throughout the entire flight envelope to the problem of fault-tolerant flight control in the presence of actuator and sensor failures requires an extensive simulation support. A simulation environment for the design, testing, and evaluation of indirect and direct adaptive flight control laws with fault-tolerant capabilities has been developed at West Virginia University (WVU) in support of the NASA Intelligent Flight Control System (IFCS) program (Boeing, 1999; Williams-Hayes, 2005). The objective of this paper is to outline the design and the development of the WVU IFCS Simulator.

The following main features are available within the simulation environment:

- selection of indirect or direct adaptive control approaches;
- selection of different PID methods within the indirect approach;
- selection of different neural augmentations within the direct approach;
- selection of different failure scenarios including selection of the failed component, type of failure, time of occurrence, and failure magnitude; and
- selection of simulation inputs/outputs.

## 2. General architecture of the simulation environment

The design of the WVU IFCS Simulator was aimed at providing a high portability, versatile, and flexible tool to address a variety of issues related to the research, development, and design of adaptive fault-tolerant flight control systems. The computational environment of choice was Matlab® and Simulink®. The flight dynamics and control (FDC) toolbox (Rauw, 1998) provides the general framework for solving the equations of motion, including atmospheric turbulence. For graphic display and pilot interaction, the dynamic model is interfaced with the aviator visual design simulator (AVDS) simulation package (Rassmussen, 2000). The aircraft dynamic model can be flown using a joystick or a set of pre-recorded command time histories. User-friendly graphical user interface (GUI) menus are used to set the conditions for the simulation scenarios.

The general structure of the WVU IFCS Simulator includes five major modules:
1 aircraft model module;
2 control system module;
3 actuator and sensor failure model;
4 failure detection and identification (FDI) schemes; and
5 user interface.

The general block diagram outlining the main modules and their interactions is shown in Figure 1.

## 3. Aircraft model module

The aircraft model module includes three major components: the wind and turbulence model, the aircraft dynamic equations of motion, and the aerodynamic database. The FDC toolbox was used to implement the Dryden model of turbulence and constant wind of pre-determined direction and magnitude. The FDC toolbox "Equations of motion solver" was modified to accommodate the computation of aerodynamic forces and moments for each control surface. This specific feature is necessary for the process of modeling the actuator failures. The aerodynamic database is structured with a number of look-up tables, which are functions of one or more dynamic variables. As an alternative, pre-trained neural networks (PTNN) acting as non-linear interpolators can be used instead.

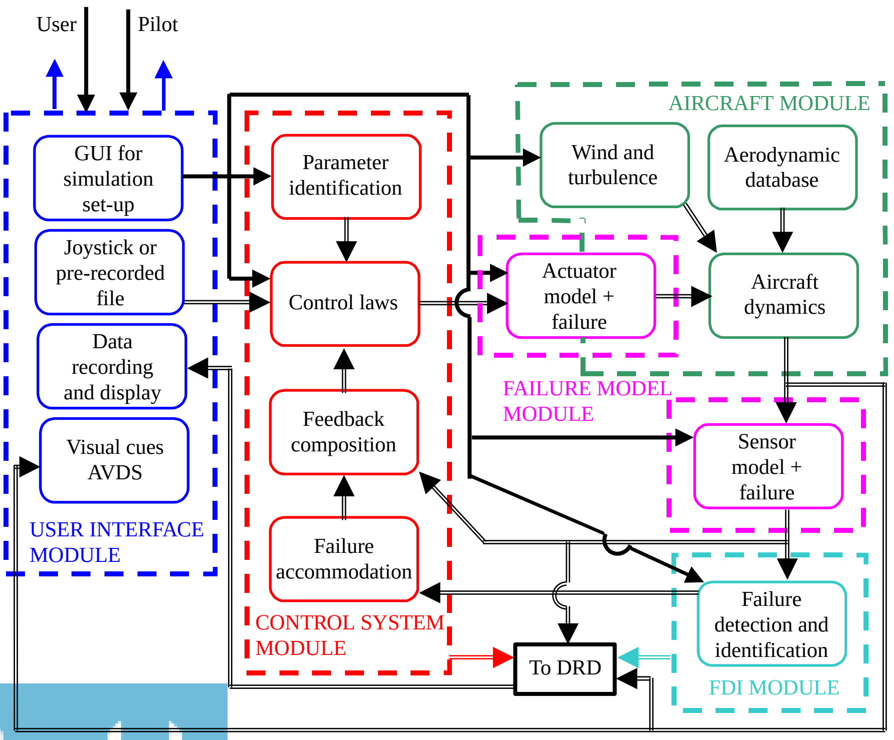
The aerodynamic model implemented within the IFCS project was derived from a non-linear model of a high-performance military aircraft distributed by NASA to academic institutions in 1990 within a student design competition (Antoniewicz *et al.*, 1988). This generic model was customized through the addition of the aerodynamics modeling of canard surfaces for the purpose of simulating the NASA IFCS F-15 research aircraft. The control laws generate commands for the differential canard, collective and differential stabilator, differential aileron, and rudder. The look-up tables were divided to represent the aerodynamic contributions from individual control surfaces – including dual rudder – to accommodate the modeling of actuator failures.

## 4. Control system module

### 4.1 Indirect adaptive control laws

The indirect adaptive control laws consist of two parts. The first part determines the commands based on parameters such as the aircraft stability and control derivatives using one of several control design techniques. The baseline implementation within WVU IFCS Simulator relies on the stochastic optimal feedforward and feedback technique (SOFFT) (Halyo *et al.*, 1992). This set of control laws has shown to provide desirable handling qualities at nominal flight conditions while retaining good performance at post-failure conditions thanks to its robustness properties (Campa *et al.*, 2004).

**Figure 1** General architecture of the WVU IFCS simulation environment



The feed forward component of the SOFFT controller is designed based on an explicit command model following approach. The resulting structure for the feed forward component is given by:

$$\bar{u} = K_x\bar{\mathbf{x}} + K_z\bar{\mathbf{z}} + K_u\bar{u}_z, \tag{1}$$

where $\bar{\mathbf{x}}$ and $\bar{\mathbf{z}}$ are the state vectors of the plant and command models, respectively, and $\bar{u}_z$ is the input to the command model.

The design of the feedback control is performed independently using a standard output feedback LQR synthesis on the feed forward plant model leading eventually to the computation of the output feedback gain matrix $\mathbf{K}_y$ (Campa *et al.*, 2004) such that:

$$\bar{u} = \mathbf{K}_y\bar{\mathbf{y}}, \tag{2}$$

where output vector $\bar{\mathbf{y}}$ includes lateral and vertical accelerations and angular rates.

Two versions of the controller have been implemented by WVU researchers: a decoupled SOFFT version (that is separate longitudinal and lateral-directional SOFFT controllers) and a full (longitudinal + lateral directional) version. The decoupled version is less computationally demanding and generates global commands, that is collective and differential deflections. The coupled version involves instead larger matrices; therefore, it is computationally more complex. It generates commands for individual control surface deflections (left and right stabilator, etc.) and hence is expected to handle better the cross-coupling after the occurrence of certain types of failures.

The second part of the controller updates the baseline parameters while the aircraft is moving throughout the flight envelope and/or abnormal flight conditions occur, such as actuator or sensor failures. The updating of the stability and control derivatives is performed through online PID techniques. A frequency domain approach – called the Fourier transform regression (FTR) method (Morelli, 1999) – was implemented within the WVU IFCS Simulator due to its very good performance and its suitability for online applications (Napolitano *et al.*, 2001).

The parameters to be estimated can be expressed either in the time domain or in the frequency domain leading, in general, to different levels of accuracy in the results (Perhinschi *et al.*, 2002b). The purpose of the PID algorithm is eventually to provide state space system matrices of the controlled plant (aircraft) that would allow updating the parameters of the SOFFT controller. In general, it is possible to identify directly the state space system matrices **A**, **B**, **C** and **D** (matrix approach) or to evaluate first the dimensionless stability and control derivatives (derivatives approach) and then use these values to compute the state space system matrices. The original PID scheme has been modified by the authors to accommodate any combination of the alternatives described above, resulting in four possibilities subject to user's choice (Perhinschi *et al.*, 2002b).

For any of these versions, depending on the flight conditions and scenario, the user can select to estimate up to 67 stability and control derivatives (including derivatives with respect to differential, global and individual – left and right – control surface deflection). A simpler version with a reduced number (13) of derivatives to be estimated is also implemented.

The updating of the SOFFT controller parameters from the PID process is triggered only when specific convergence criteria of the PID process are met. These criteria are based on the time histories of the estimates and the standard deviations of the estimation errors as computed using the FTR algorithm (Morelli, 1999). The convergence criteria require that the time histories stay within a given bound around the average for a specified time window. The width of the bound and that of the time window are imposed by the user. The user can also select between convergence criteria for the aerodynamic parameters (that is any of the 67 stability and control derivatives), for the standard deviation of the estimation error for each of the parameters, or both (Perhinschi *et al.*, 2002a). The updating of the SOFFT controller parameters can also be performed using an online neural network (OLNN), which in turn is trained using the output of the PID process as long as the convergence conditions are met.

The block diagram of the indirect adaptive control laws is shown in Figure 2. Note that the "Derivative Update" block includes the OLNN and the PID convergence test.

### 4.2 Direct adaptive control laws

The direct adaptive control laws – as shown in Figure 3 – use a "model following" architecture based on non-linear dynamic inversion (NLDI) augmented with artificial neural networks (NNs). This architecture has shown capabilities for compensating tracking errors while internal parameters including the NN gains and output remain bounded (Calise and Sharma, 2000).

Within the NLDI control laws, pilot stick and pedal displacements are first converted into angular rate commands. Next, first and second order reference models are used to determine the desired aircraft response in terms of angular rates and their derivatives such that Level 1 handling qualities are ensured (US Department of Defense, 1980). The tracking errors with respect to these desired angular rates are used to provide proportional, integral, and derivative compensation. Online learning NNs generate augmentation commands to compensate for inversion errors. Pseudo-controls in terms of angular acceleration commands $(\dot{p}_c, \dot{q}_c, \dot{r}_c)$ are computed based on pilot input, tracking error compensation, and NN output and used to obtain, through NLDI, generic moment producing deflections on the three axes:

$$\begin{bmatrix} \delta_{a_{\mathrm{com}}} \\ \delta_{e_{\mathrm{com}}} \\ \delta_{r_{\mathrm{com}}} \end{bmatrix} = \mathbf{B}^{-1} \begin{bmatrix} \dot{p}_c - L_1 \\ \dot{q}_c - M_1 \\ \dot{r}_c - N_1 \end{bmatrix}, \tag{3}$$

$L_1$, $M_1$ and $N_1$ are the non-linear terms of the moment equations and **B** is the state space system control matrix computed at one particular flight condition. Finally, the control surface actual deflections are computed from $\delta_{a_{\mathrm{com}}}$, $\delta_{e_{\mathrm{com}}}$, $\delta_{r_{\mathrm{com}}}$ according to a control allocation algorithm. First and second order actuator dynamics models are also included. The elements of matrix **B** can be kept constant or can be updated using look-up tables or PTNNs (Perhinschi *et al.*, 2004).

**Figure 2** Block diagram of the indirect adaptive control laws

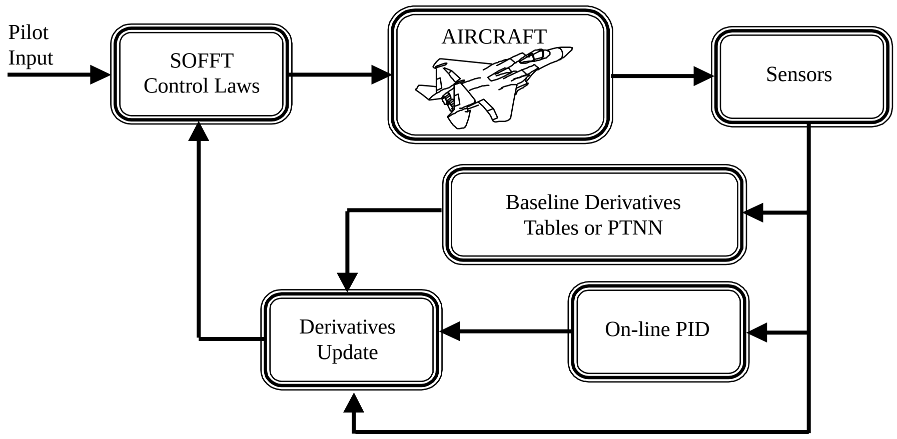
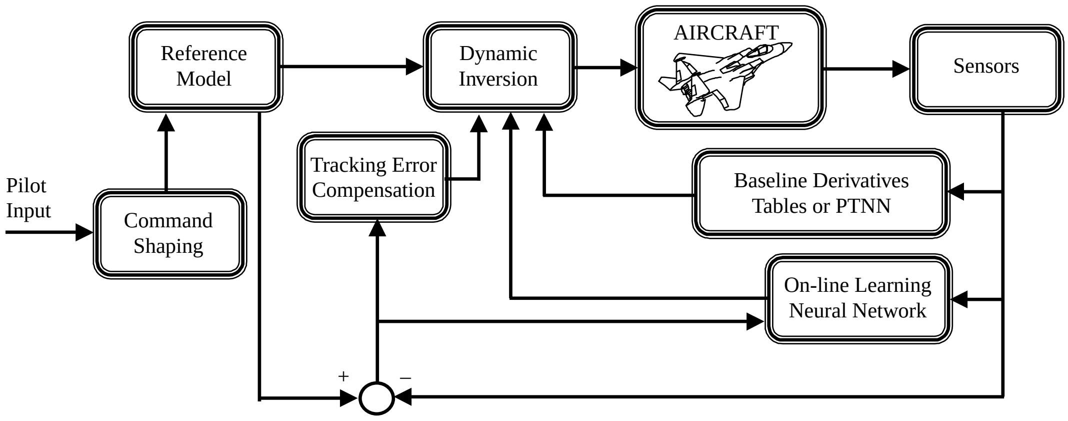


**Figure 3** Block diagram of the direct adaptive control laws


Reference Model
Dynamic Inversion
AIRCRAFT
Sensors
Pilot Input
Command Shaping
Tracking Error Compensation
Baseline Derivatives Tables or PTNN
On-line Learning Neural Network
+
−

Three different types of online learning NNs have been implemented within the WVU IFCS Simulator: the extended minimal resource allocation network – EMRAN (Lu *et al.*, 2000), SigmaPi (Shin and Ghosh, 1991), and the single hidden layer – SHL NN (Calise *et al.*, 1998).

The EMRAN is an improved radial basis function network featuring a growing and pruning mechanism allowing the allocation of additional neurons in regions of the state space where the mapping accuracy is poor while avoiding the excessive growth of the network. Only the parameters of the most activated neurons are updated; therefore, the computational effort is minimized for this class of neural augmentation. For Gaussian basis functions, the estimate is computed with the expression:

$$\hat{y}(\mathbf{x}, \theta) = \sum_{i=1}^{M} w_i e^{|\mathbf{x}-\mu_i|^2/2\sigma_i^2}, \tag{4}$$

where $\mathbf{x}$ is the input vector, $\theta$ is the set of parameters to be tuned by the learning algorithm including the weight $w$, the Gaussian center positions $\mu$, and the variances $\sigma$. A new neuron is initiated if three distinct criteria are simultaneously satisfied: the estimation error, the windowed estimation error, and the distance from the input to the nearest center must be larger than selected thresholds. When one of the criteria is not met, the tuning parameters are updated using the relationship:

$$\theta(k+1) = \theta(k) - \eta \frac{\partial \hat{y}(k)}{\partial \theta(k)}\bigg|_{(k)} \cdot e(k), \tag{5}$$

where $e(k)$ is the estimation error and $\eta$ is the learning rate.

The SigmaPi NN is equivalent to an adaptive polynomial mapping. Inputs to the networks are pseudo control acceleration commands, bias terms, and sensor feedback. For each channel, a first neural layer output $\{c_i\}$ is computed as functions of inputs and previous-step NN outputs. Next, these variables are multiplied to each other and eventually the output of the NN is computed as the weighted sum of these products:

$$U_{\mathrm{NN}} = w^{\mathrm{T}} f(\{c_i\}), \tag{6}$$

where $f$ is computed from the intermediate outputs $\{c_i\}$ using a nested Kronecker product. The network weights are determined using the following adaptation law based on a modified delta rule:

$$\dot{w} = -G(U_e \cdot f + L|U_e|w), \tag{7}$$

$G$ and $L$ are user selected specific gains.

The SHL NN is using a classic sigmoid activation function of the form:

$$\sigma(\xi) = \frac{1}{1 + \mathrm{e}^{-a\xi}}. \tag{8}$$

The output of the network is given by the relationship:

$$y_i = \sum_{j=1}^{m}\left[w_{ij}\sigma\left(\sum_{k=1}^{n} v_{jk}x_k + \theta_{vj}\right) + \theta_{wi}\right], \quad i = 1, 2, \ldots, p, \tag{9}$$

where $W = \{w_{ij}\}$ are the interconnection weights between the hidden and the output layer, $V = \{v_{jk}\}$ are the interconnection weights between the input and the hidden layer, and $\theta_{vj}$, $\theta_{wi}$ are bias terms. Based on a Lyapunov analysis, boundedness of the weights and the output of the NN is ensured provided the updating laws are given by:

$$\begin{aligned}\dot{W} &= -\gamma_W\left[(\sigma - \sigma' V^{\mathrm{T}}x)e_x^{\mathrm{T}} + \lambda_W\|e_x\|W\right]\\ \dot{V} &= -\gamma_V\left[xe_x^{\mathrm{T}}W^{\mathrm{T}}\sigma + \lambda_V\|e_x\|V\right],\end{aligned} \tag{10}$$

where $e_x$ are state errors and $\gamma_W, \gamma_V, \lambda_W, \lambda_V$ are design parameters (learning rates).

### 4.3 Failure accommodation

#### *4.3.1 Actuator failure accommodation*

Both the indirect and the direct adaptive control laws have inherent accommodation capabilities to deal with the occurrence of actuator failures. An actuator failure induces abrupt changes in the dynamic characteristics of the system, which are equivalent to changes of the stability and control derivatives. The PID process, as part of the indirect adaptive control laws, has the objective to detect such changes and to update the baseline parameters. In other words, following a failure, the controlled plant changes but so do the control laws. The direct adaptive control laws implemented within the WVU simulation environment are based on NLDI with parameters at one reference flight condition. Errors may occur from three sources: modeling errors, errors due to different flight conditions as the aircraft moves within the flight envelope, and errors due to dynamic changes following actuator failures. These errors are expected to be cancelled by additional commands generated by artificial NNs. The online learning NNs compensate for modeling errors and effects of actuator failures while the PTNNs compensate for departure from reference flight conditions.

#### *4.3.2 Sensor failure accommodation*

The accommodation for sensor failures is accomplished in the same way for both categories of control laws. Although, failures of a larger variety of sensors have been implemented, the failure accommodation scheme only addresses failures of the angular rates sensors. On each channel, online learning NNs are used to generate estimates of the angular rates from inputs including angle of attack, sideslip angle, control surface deflections, and off-axis angular rates. These NNs are part of a sensor failure detection scheme that will be described in more detail in Section 6. Once a sensor failure has been identified, the respective NN stops learning and its output is fed into the control laws replacing the output of the faulty sensor.

## 5. Actuator and sensor failure modeling

### 5.1 Actuator failure modeling

Two types of control surface failure have been modeled and implemented within the simulation environment. The first failure type corresponds to an actuator mechanism failure and results in a locked surface; in fact, at the failure occurrence, the control surface remains fixed in the current or in a user prescribed position. The second failure type corresponds to a physical destruction and/or deformation of the control surface. It consists of a deterioration of the aerodynamic "efficiency" of the control surface starting at the occurrence of the failure. The user can select different failure parameters such as type, occurrence time, position, and magnitude. Any of the individual eight control surfaces of the baseline aircraft – which features canards and dual fin rudder – may be subjected to a failure, that is left or right stabilators, ailerons, canards, and rudders.

A failure involving a blockage of the control surface at a fixed deflection does not alter the aerodynamic properties of the control surface. However, each surface in a pair (left and right) will have different deflections and the resulting moments and forces must be computed individually. Therefore, the aerodynamic look-up tables must be divided such that the contributions of each individual control surface are isolated.

A control failure that involves physical destruction of the control surface may alter the aerodynamic properties in manners that can be both qualitative (affecting the nature of the aerodynamic phenomena involved) and quantitative (affecting the magnitude of characteristic parameters). In modeling this type of failure, it was assumed that the alteration of aerodynamic properties is such that the forces and moments generated by the control surface after the failure differ from those before the failure by a proportional factor $(\bar{s}_d)$ affecting an efficiency parameter $E_{u_k}$, $k = 1, \ldots, m$, with $m$ the total number of control surfaces.

Let the aerodynamic forces and moments be expressed as functions of the aircraft states $x$, their derivatives, and the inputs $u$ as:

$$\overline{\mathrm{FM}} = \bar{f}(\bar{x}, \dot{\bar{x}}, \bar{u}). \tag{11}$$

Through appropriate manipulation of the aerodynamic models, the forces and moments can be expressed by separating various contributions:

$$\overline{\mathrm{FM}} = \bar{f}_{x\,\mathrm{WB}}(\bar{x}, \dot{\bar{x}}) + \bar{f}_{xC}(\bar{x}, \dot{\bar{x}}) + \bar{f}_{xu}(\bar{x}, \dot{\bar{x}}, \bar{u}) + \bar{f}_u(\bar{u}), \tag{12}$$

where $\bar{f}_{x\,\mathrm{WB}}$ represents the contribution of the wing and fuselage to the forces and moments while $\bar{f}_{xC}$, $\bar{f}_{xu}$, and $\bar{f}_u$ represent the contribution of the control surfaces. Furthermore, the contributions from the $m$ control surfaces can be expressed as:

$$\begin{aligned}\overline{\mathrm{FM}} = \bar{f}_{x\,\mathrm{WB}}(\bar{x}, \dot{\bar{x}}) + \sum_{k=1}^{m}(\bar{f}_{xC_k}(\bar{x}, \dot{\bar{x}}, E_{u_k}) + \bar{f}_{xu}(\bar{x}, \dot{\bar{x}}, u_k, E_{u_k})\\ + \bar{f}_u(u_k, E_{u_k})),\end{aligned} \tag{13}$$

where $E_{u_k}$ are efficiency parameters that must be defined/identified for each control surface. For example, in the case of the elevator or stabilator, the efficiency parameters are selected to be the derivatives of the aerodynamic normal force with respect to left and right elevator deflection, respectively, $C_{Z\delta_{eL}}$ and $C_{Z\delta_{eR}}$.

After a control failure involving the physical destruction of the control surface the expression of the forces and moments acting on the aircraft can be expressed using:

$$\overline{\mathrm{FM}} = \bar{f}_{x\,\mathrm{WB}}(\bar{x},\dot{\bar{x}}) + \sum_{k=1}^{m}\left(\bar{f}_{xC_k}(\bar{x},\dot{\bar{x}},s_{d_k}E_{u_k}) + \bar{f}_{xu}(\bar{x},\dot{\bar{x}},u_k,s_{d_k}E_{u_k}) + \bar{f}_u(u_k,s_{d_k}E_{u_k})\right), \quad (14)$$

where the surface damage parameter $s_d$ models the magnitude of the failure through the ratio between the efficiency parameter after and before the failure occurring moment:

$$s_{d_k} = \frac{(E_{u_k})_{\mathrm{AfterFailure}}}{(E_{u_k})_{\mathrm{BeforeFailure}}}. \quad (15)$$

Therefore, $s_d \in [0,1]$, with $s_d = 1$ for the "no failure" case and $s_d = 0$ for a failure involving a completely missing surface or a complete loss of "efficiency". The terms $\bar{f}_{xC_k}, \bar{f}_{xu}$ and $\bar{f}_u$ in equation (14) are all affected by the failure. These terms are expressed in terms of stability and control derivatives, which in turn depend on the efficiency parameters. Finally, the separation of the contributions of the control surfaces (left and right) to the derivatives allows the computation of forces and moments at post-failure conditions as functions of the efficiency parameters. More details on the general modeling of the actuator failures are presented by Perhinschi *et al.* (2006a).

### 5.2 Sensor failure modeling

All the sensors that are typically used in the control laws feedback, such as angular rate, aerodynamic angles of attack, altitude, and speed sensors have been first modeled as simple first order systems:

$$\sigma_a(s) = \frac{1}{\tau_s s + 1}\,x(s), \quad (16)$$

where $(\sigma_a)$ is a noiseless sensor output and $x$ is the true output as it results from the mathematical model of the aircraft.

The sensor modeling, to include various types of failure, will consider the following general expression for the measurement:

$$\sigma_m(t) = \max[\min(\bar{K}_s(t)\cdot\sigma_a(t) + \bar{\sigma}_b(t) + \bar{K}_d(t)\cdot(t - t_{\mathrm{f}}), \sigma_{\max}), \sigma_{\min}] + \bar{K}_n(t)\cdot\sigma_n(t), \quad (17)$$

where, $\bar{K}_s, \bar{K}_d, \bar{K}_n,$ and $\bar{\sigma}_b$ are, respectively, a sensor output scaling factor, a drift factor, a noise amplifier, and a bias. In general, the sensor output and the noise scaling factors are defined by the following relationship, where $t_{\mathrm{f}}$ is the moment of failure occurrence:

$$\bar{K}_i(t) = \begin{cases} 1 & \text{for} \quad t < t_{\mathrm{f}} \\ K_i & \text{for} \quad t \geq t_{\mathrm{f}} \end{cases}, \quad i = s \text{ or } n. \quad (18)$$

The drift factor is defined as:

$$\bar{K}_d(t) = \begin{cases} 0 & \text{for} \quad t < t_{\mathrm{f}} \\ K_d & \text{for} \quad t \geq t_{\mathrm{f}} \end{cases} \quad (19)$$

The bias of the sensor output $\bar{\sigma}_b$ can be reached faster or slower over a time interval $\Delta t$ and is defined as:

$$\bar{\sigma}_b(t) == \begin{cases} 0 & \text{for} \quad t < t_{\mathrm{f}} \\ \frac{\sigma_b}{\Delta t}(t - t_{\mathrm{f}}) & \text{for} \quad t_{\mathrm{f}} \leq t < t_{\mathrm{f}} + \Delta t \\ \sigma_b & \text{for} \quad t \geq t_{\mathrm{f}} + \Delta t \end{cases} \quad (20)$$

By properly selecting values for the eight modeling parameters ($K_s$, $K_d$, $K_n$, $\Delta t$, $\sigma_b$, $\sigma_{\max}$, $\sigma_{\min}$, and $t_{\mathrm{f}}$) the following types of sensor failures can be simulated:

- biased sensor output with variable rate;
- drifting output;
- constant or saturated output; and
- increased output noise.

For example, normal operation corresponds to $K_s = 1, K_d = 0,$ $K_n = 1$, and $\sigma_b = 0$. A roll rate gyro biased by 10°/s reaching this bias over 1.5 s is modeled if $K_s = 1$, $K_d = 0$, $K_n = 1$, $\sigma_b = 10$, and $\Delta t = 1.5$.

## 6. Failure detection and identification

The actuator and sensor FDI scheme was designed to address failures on any of the four aerodynamic control paired surfaces and on any of the angular rate sensors. Without any loss of generality, the FDI scheme can be modified to deal with different actuator and/or sensor failures. The scheme is based on the use of neural estimators interfaced with correlation functions of the aircraft angular rates.

Particular emphasis is placed on the differentiation between sensor and actuator failures. Within each category, the scheme can determine which element has failed. Specifically, for an actuator failure, the scheme can specify whether it is a stabilator, aileron, canard, or rudder failure whereas in the event of a sensor failure the scheme needs to correctly detect and isolate the failed sensor.

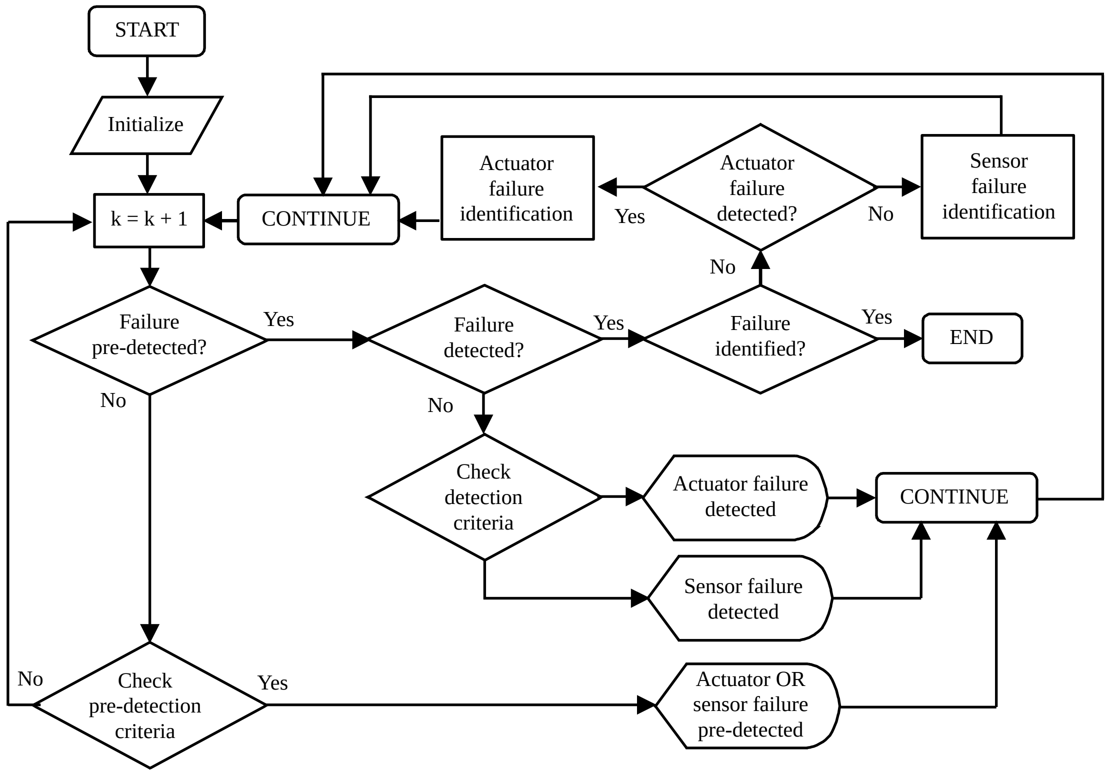
The general actuator and sensor FDI scheme can be divided into the following tasks:

- preliminary detection (a failure of an unspecified primary control surface or sensor is detected);
- detection (the failure is associated to a primary control surface or sensor);
- preliminary identification (this phase applies only to control surface failures and consists of evaluating the failure to be a longitudinal- or lateral-directional failure); and
- identification (if previously detected as a control surface failure, the failure is identified to be a stabilator, canard, aileron, or rudder failure; if previously detected as a sensor failure, it is identified to be a roll, pitch, or yaw rate sensor failure).

The high-level logical block diagram of the actuator and sensor FDI scheme is shown in Figure 4.

The actuator and sensor FDI scheme relies primarily on two sets of NNs. The first one, referred to as the main neural network (MNN), generates estimates of the angular rates ($\hat{p}_{\mathrm{MNN}}(k)$, $\hat{q}_{\mathrm{MNN}}(k)$, and $\hat{r}_{\mathrm{MNN}}(k)$) at time $k$, using measurements from time instant $k-1$ to $k-m$. The inputs to the MNNs include all gyro measurements, angle of attack, sideslip angle and control surface deflections. The second set, the decentralized neural networks (DNNs), also generates angular rate estimates ($\hat{p}_{\mathrm{DNN}}(k)$, $\hat{q}_{\mathrm{DNN}}(k)$, and $\hat{r}_{\mathrm{DNN}}(k)$) but

**Figure 4** Integrated sensor and actuator FDI scheme


START
Initialize
k = k + 1
CONTINUE
Actuator failure identification
Yes
Actuator failure detected?
No
Sensor failure identification
No
Failure pre-detected?
Yes
Failure detected?
Yes
Failure identified?
Yes
END
No
No
Check detection criteria
Actuator failure detected
CONTINUE
Sensor failure detected
No
Check pre-detection criteria
Yes
Actuator OR sensor failure pre-detected

does not use as an input measurements from the respective gyro. For both the MNNs and DNNs, combinations of an ADALINE and EMRAN NN working in parallel have been implemented on each channel. This approach was adopted to achieve desirable performance in the presence of large non-linearities – using the adaptation capabilities of the EMRAN NNs – without the computational burden on operation in areas with reduced non-linearities – using the simpler ADALINE NNs.

Within the scheme, the estimates from the MNN are compared with the actual measurements at time $k$ ($p(k)$, $q(k)$, and $r(k)$) to define the main quadratic estimation error (MQEE) parameter:

$$\mathrm{MQEE}(k) = \frac{1}{2}[(p(k) - \hat{p}_{\mathrm{MNN}}(k))^2 + (q(k) - \hat{q}_{\mathrm{MNN}}(k))^2 + (r(k) - \hat{r}_{\mathrm{MNN}}(k))^2]. \tag{21}$$

Additionally, a NN output quadratic estimation error (OQEE) parameter is defined by comparing the estimates from the MNN and the individual DNNs:

$$\mathrm{OQEE}(k) = \frac{1}{2}[(\hat{p}_{\mathrm{DNN}}(k) - \hat{p}_{\mathrm{MNN}}(k))^2 + (\hat{q}_{\mathrm{DNN}}(k) - \hat{q}_{\mathrm{MNN}}(k))^2 + (\hat{r}_{\mathrm{DNN}}(k) - \hat{r}_{\mathrm{MNN}}(k))^2]. \tag{22}$$

Furthermore, for each channel, a decentralized quadratic estimation error (DQEE) parameter is computed as the difference between the outputs of the DNN and the actual measurements:

$$\mathrm{DQEE}_x(k) = \frac{1}{2}(\hat{x}_{\mathrm{DNN}}(k) - x(k))^2 \quad \text{where} \quad x = p, q, r. \tag{23}$$

These parameters, individually and in weighted combinations with the correlation functions of the angular rates, are used to formulate criteria for all the four phases of the actuator and sensor FDI process based on constant and floating thresholds. More details on the development and performance of the FDI scheme are presented by Napolitano *et al.* (2000) and Perhinschi *et al.* (2006b, 2007).

## 7. User interface

### 7.1 General simulation setup

A set of user-friendly GUI menus are designed for setting up the simulation scenarios. The information to be provided through these menus can be organized in three main categories, that is the control laws setup, the simulation input module, and the output module, as shown in Figure 5.

### 7.2 Control laws setup module

For both the indirect and direct adaptive control laws, the user can select to simulate the following combinations of structural components:

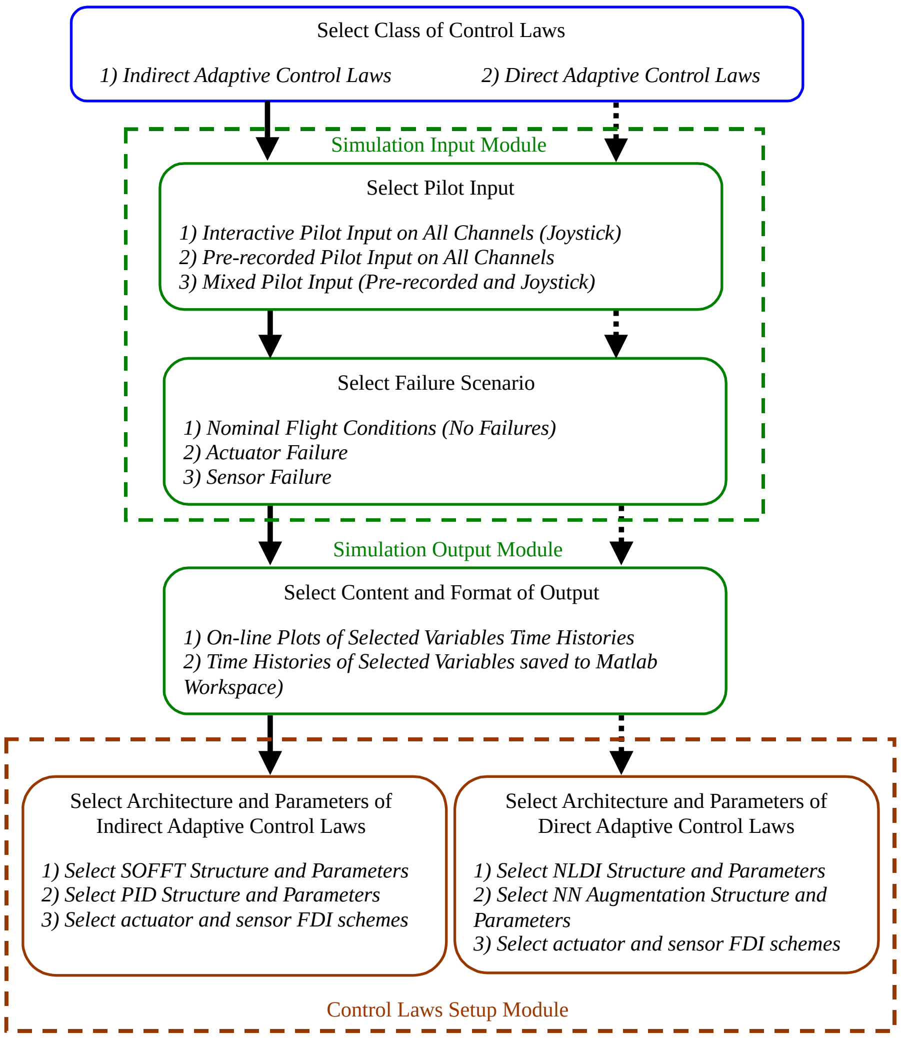
- control laws only;
- control laws and actuator FDI scheme;
- control laws and sensor FDI scheme; and
- control laws and integrated actuator and sensor FDI scheme with or without sensor failure accommodation.

The parameters of the FDI schemes are provided through a Matlab script file.

#### *7.2.1 Indirect adaptive control laws*

The user can select among several alternative architectures allowing to focus on specific research issues. For example, one

**Figure 5** General structure of the user interface


Select Class of Control Laws
1) Indirect Adaptive Control Laws
2) Direct Adaptive Control Laws
Simulation Input Module
Select Pilot Input
1) Interactive Pilot Input on All Channels (Joystick)
2) Pre-recorded Pilot Input on All Channels
3) Mixed Pilot Input (Pre-recorded and Joystick)
Select Failure Scenario
1) Nominal Flight Conditions (No Failures)
2) Actuator Failure
3) Sensor Failure
Simulation Output Module
Select Content and Format of Output
1) On-line Plots of Selected Variables Time Histories
2) Time Histories of Selected Variables saved to Matlab Workspace)
Select Architecture and Parameters of Indirect Adaptive Control Laws
1) Select SOFFT Structure and Parameters
2) Select PID Structure and Parameters
3) Select actuator and sensor FDI schemes
Select Architecture and Parameters of Direct Adaptive Control Laws
1) Select NLDI Structure and Parameters
2) Select NN Augmentation Structure and Parameters
3) Select actuator and sensor FDI schemes
Control Laws Setup Module

scenario features the aircraft at open-loop conditions connected to the PID block such that PID issues can be isolated and investigated. Another scenario includes the SOFFT controller featuring real time parameter updating based on PID results once specific convergence criteria are met. An online learning NN can also be used to support the updating of the controller parameters at post-failure conditions.

Within the indirect adaptive control laws class, the user can select among the following alternatives:

- decoupled or coupled SOFFT control laws;
- frequency or time domain representation for the PID parameters;
- matrix approach or derivatives approach for the PID algorithm;
- variable list of parameters to be estimated and updated within the control laws; and
- several PID convergence criteria and associated parameters.

Examples of the interactive menus developed for the setup of the indirect adaptive control laws are shown in Figure 6.

*7.2.2 Direct adaptive control laws*

Within the direct adaptive control laws class, the user can select to maintain the parameters of the NLDI fixed at a default initial flight condition or to update these parameters at specified time intervals. The updated values can be computed online or provided by PTNNs. The default initial flight condition can be modified in the initial data Matlab script file. Depending on the type of neural augmentation, the alternative configurations consist of:

- NLDI only.
- NLDI and EMRAN NN.
- NLDI and SigmaPI NN.
- NLDI and SHL NN.

Examples of the interactive menus developed for the setup of the direct adaptive control laws structure are shown in Figure 7.

### 7.3 Simulation input module

The pilot input on each of the four control channels (longitudinal, lateral, directional, and throttle) can be selected to be provided online using the joystick or as pre-recorded (or otherwise built) stick displacement time histories. The pilot input menu is shown in Figure 8.

From the main control laws setup menus (Figures 6 and 7), the user can select between nominal and failure flight conditions. If failure conditions are selected, then it must be specified whether

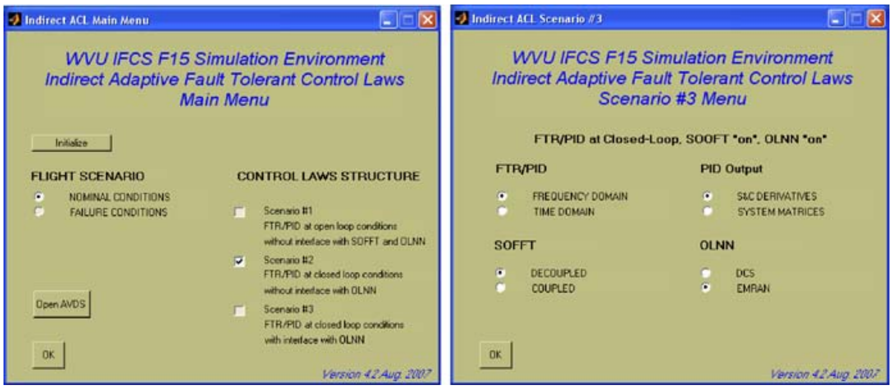


**Figure 6** Setup menus for the structural characteristics of the indirect adaptive control laws

Indirect ACL Main Menu

WVU IFCS F15 Simulation Environment
Indirect Adaptive Fault Tolerant Control Laws
Main Menu

FLIGHT SCENARIO: [✓ NOMINAL CONDITIONS] FAILURE CONDITIONS

CONTROL LAWS STRUCTURE:
- Scenario #1 FTR/PID at open loop conditions without interface with SOFFT and OLNN
- [✓ Scenario #2 FTR/PID at closed loop conditions without interface with OLNN]
- Scenario #3 FTR/PID at closed loop conditions with interface with OLNN

Version 4.2 Aug. 2007

Indirect ACL Scenario #3

WVU IFCS F15 Simulation Environment
Indirect Adaptive Fault Tolerant Control Laws
Scenario #3 Menu

FTR/PID at Closed-Loop, SOOFT "on", OLNN "on"

FTR/PID: [✓ FREQUENCY DOMAIN] TIME DOMAIN
PID Output: [✓ S&C DERIVATIVES] SYSTEM MATRICES
SOFFT: [✓ DECOUPLED] COUPLED
OLNN: DCS [✓ EMRAN]

Version 4.2 Aug. 2007

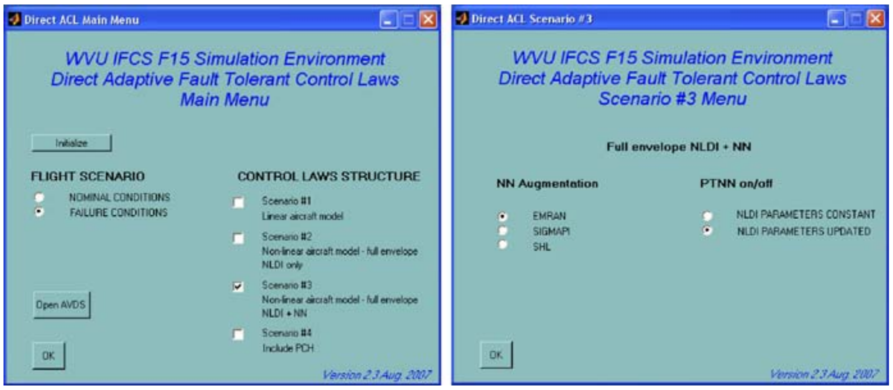


**Figure 7** Setup menus for the structural characteristics of the direct adaptive control laws

Direct ACL Main Menu

WVU IFCS F15 Simulation Environment
Direct Adaptive Fault Tolerant Control Laws
Main Menu

FLIGHT SCENARIO: NOMINAL CONDITIONS [✓ FAILURE CONDITIONS]

CONTROL LAWS STRUCTURE:
- Scenario #1 Linear aircraft model
- Scenario #2 Non-linear aircraft model - full envelope NLDI only
- [✓ Scenario #3 Non-linear aircraft model - full envelope NLDI + NN]
- Scenario #4 Include PCH

Version 2.3 Aug. 2007

Direct ACL Scenario #3

WVU IFCS F15 Simulation Environment
Direct Adaptive Fault Tolerant Control Laws
Scenario #3 Menu

Full envelope NLDI + NN

NN Augmentation: [✓ EMRAN] SIGMAPI SHL
PTNN on/off: NLDI PARAMETERS CONSTANT [✓ NLDI PARAMETERS UPDATED]

Version 2.3 Aug. 2007

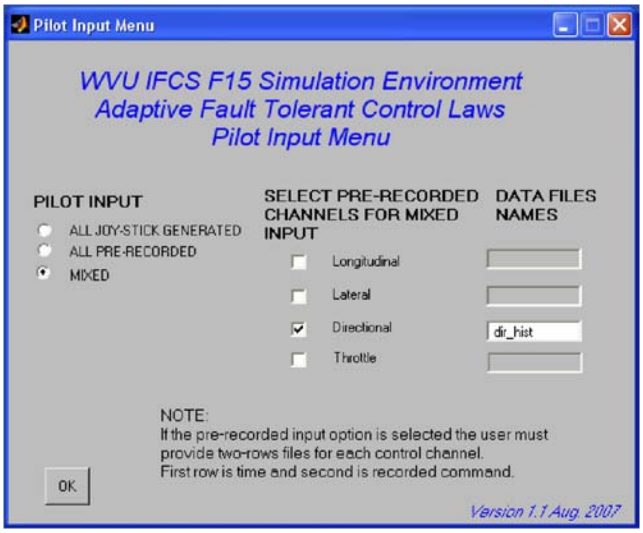


**Figure 8** Pilot input setup menu

Pilot Input Menu

WVU IFCS F15 Simulation Environment
Adaptive Fault Tolerant Control Laws
Pilot Input Menu

PILOT INPUT: ALL JOY-STICK GENERATED, ALL PRE-RECORDED, [✓ MIXED]

| SELECT PRE-RECORDED CHANNELS FOR MIXED INPUT | DATA FILES NAMES |
|---|---|
| Longitudinal | |
| Lateral | |
| [✓ Directional] | dir_hist |
| Throttle | |

NOTE:
If the pre-recorded input option is selected the user must provide two-rows files for each control channel.
First row is time and second is recorded command.

Version 1.1 Aug. 2007

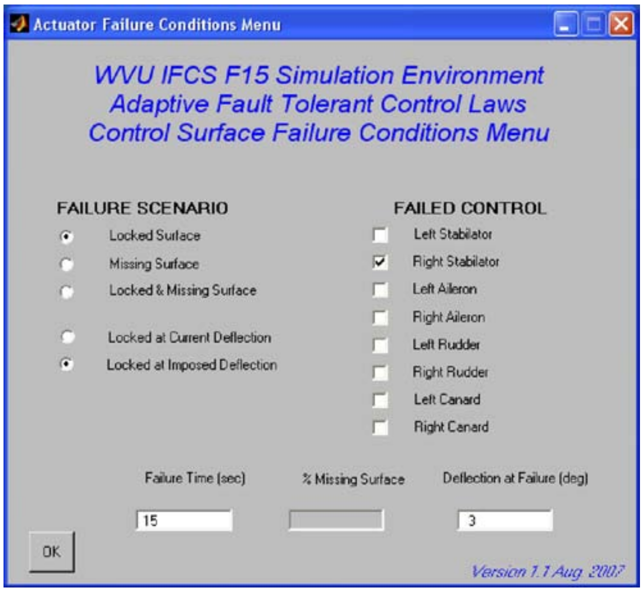


**Figure 9** Setup menu for actuator failure type and parameters

Actuator Failure Conditions Menu

WVU IFCS F15 Simulation Environment
Adaptive Fault Tolerant Control Laws
Control Surface Failure Conditions Menu

FAILURE SCENARIO: [✓ Locked Surface] Missing Surface, Locked & Missing Surface; Locked at Current Deflection [✓ Locked at Imposed Deflection]

FAILED CONTROL: Left Stabilator, [✓ Right Stabilator], Left Aileron, Right Aileron, Left Rudder, Right Rudder, Left Canard, Right Canard

| Failure Time (sec) | % Missing Surface | Deflection at Failure (deg) |
|---|---|---|
| 15 | | 3 |

Version 1.1 Aug. 2007

the failure affects a primary control surface or a sensor. Should the user select actuator failure conditions, the user can set the different failure parameters in the control surface failure conditions menu (Figure 9). In fact, in this menu the user can select among failure on different control surfaces and among jammed control, missing surface or both; furthermore, jammed in current position or in a user-selected position. Additional failure parameters include failure occurrence time, percentage missing, and jamming position. Should the user select sensor failure conditions, the user is then directed to another interactive menu to select the particular gyro affected by the failure, the type of failure, and the moment of occurrence.

### 7.4 Simulation output module

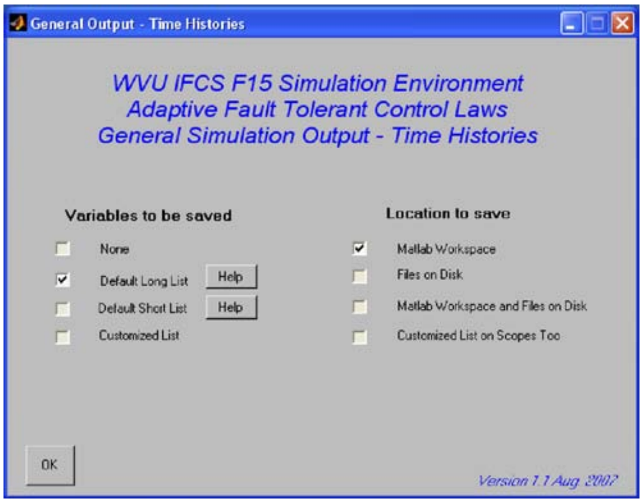
The user has the following options to handle the simulation outputs:

- time histories for the variables of interest (saved to Matlab workspace or to files on disk);
- visualization of variables during or after the simulation using Simulink scopes; and
- pop-up warnings for inconsistent user input values and as a result of FDI.

The user can organize the information to be saved for post-simulation processing and analysis using the menu shown in Figure 10. From the visualization menu shown in Figure 11, the user can establish the variables to be monitored during the simulation. This menu is kept active throughout the simulation allowing visualization of variables after the simulation as well.

The menus in Figures 10 and 11 are essentially gateways to additional options. In general, the user is given the opportunity to access all relevant parameters such as the aircraft state and control variables, the output and parameters of the control laws, the pilot input and its intermediate processing within the control laws, the tracking errors and their statistics, the S&AFDI parameters, the estimated and actual values of the stability and control derivatives (for indirect adaptive architectures), NN inputs, outputs, and adaptive weights (for direct adaptive architectures).

**Figure 10** Setup of time histories output



General Output - Time Histories

WVU IFCS F15 Simulation Environment
Adaptive Fault Tolerant Control Laws
General Simulation Output - Time Histories

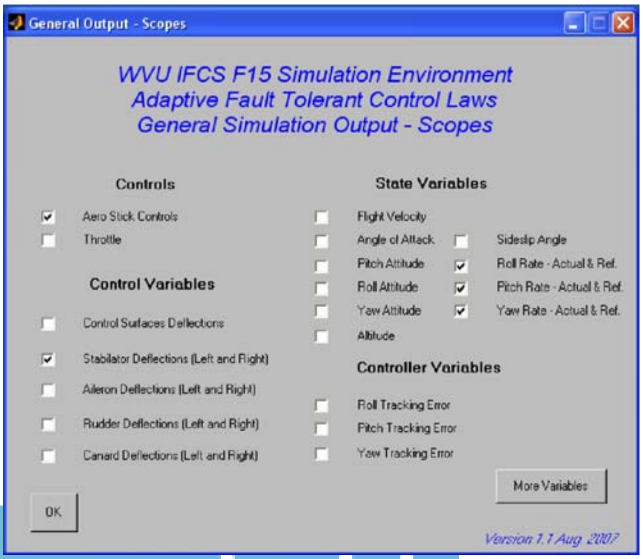
Variables to be saved
- [ ] None
- [✓] Default Long List (Help)
- [ ] Default Short List (Help)
- [ ] Customized List

Location to save
- [✓] Matlab Workspace
- [ ] Files on Disk
- [ ] Matlab Workspace and Files on Disk
- [ ] Customized List on Scopes Too

OK

Version 1.1 Aug 2007

**Figure 11** Setup of simulation output visualization



General Output - Scopes

WVU IFCS F15 Simulation Environment
Adaptive Fault Tolerant Control Laws
General Simulation Output - Scopes

Controls
- [✓] Aero Stick Controls
- [ ] Throttle

Control Variables
- [ ] Control Surfaces Deflections
- [✓] Stabilator Deflections (Left and Right)
- [ ] Aileron Deflections (Left and Right)
- [ ] Rudder Deflections (Left and Right)
- [ ] Canard Deflections (Left and Right)

State Variables
- [ ] Flight Velocity
- [ ] Angle of Attack
- [ ] Sideslip Angle
- [ ] Pitch Attitude
- [✓] Roll Rate - Actual & Ref.
- [ ] Roll Attitude
- [✓] Pitch Rate - Actual & Ref.
- [ ] Yaw Attitude
- [✓] Yaw Rate - Actual & Ref.
- [ ] Altitude

Controller Variables
- [ ] Roll Tracking Error
- [ ] Pitch Tracking Error
- [ ] Yaw Tracking Error

More Variables

OK

Version 1.1 Aug 2007

Figures 12 and 13 show the general graphics of the simulation environment, which are based on AVDS views and Simulink scopes for two distinct cases. Figure 12 shows results from a simulation at nominal flight conditions with the indirect adaptive control laws featuring decoupled SOFFT control laws, time domain derivations, and stability and control derivatives estimation. The purpose of this test was to analyze the performance and convergence of the PID process, which is one of the most important issues related to this type of control laws architecture. For example, the scope at top left shows the variation of the estimated values of $C_{Z\alpha}$, which is the non-dimensional derivative of the aerodynamic force along the vertical aircraft body axis with respect to the angle of attack. This parameter plays an important role within the control laws. It can be seen that after a transient of about 3 s the estimate converges to the actual value – about $-4$ – of the derivative.

Results from a simulation with direct adaptive control laws featuring NLDI augmented with SHL NN without PTNN updating are shown in Figure 13. A left stabilator failure occurs at $t = 20$ s when the control surface is locked at trim $+8°$. The purpose of this test was to analyze the performance of the actuator FDI scheme and the general failure accommodation capabilities of the direct adaptive control laws.

The scopes bar in the middle of Figure 13 shows the variation of FDI parameters based on correlations of aircraft angular rates. It can be seen that, immediately after the occurrence of the failure, the detectors based on correlation between pitch rate and both roll and yaw rate experience significant increase as a result of coupling between channels, which is an important fingerprint of a stabilator failure. In fact, the FDI scheme detects and correctly identifies the failure as a stabilator failure and produces the pop-up warning shown at the right bottom of the Figure 13. The control laws are designed to follow the output of an ideal reference model for optimal handling qualities; therefore, the errors in tracking the output of this reference model are the metric for control laws performance evaluation. The scopes at the left side of Figure 13 show the reference model output and the actual values of the angular rates. There is an excellent match before the failure. A significant discrepancy occurs at the moment of the failure. After a short transient, the NN augmentation is capable to "learn" the new operational conditions and provide the necessary compensation to maintain good tracking performance.

## 8. Conclusions

A Matlab/Simulink-based simulation environment has been developed for the design and testing of indirect and direct adaptive flight control laws with fault tolerant capabilities to detect, identify, and accommodate failures of aircraft actuators and sensors.

Indirect adaptive flight control laws are implemented based on an optimal controller and frequency domain online

**Figure 12** WVU IFCS simulation environment graphics – indirect adaptive flight control laws

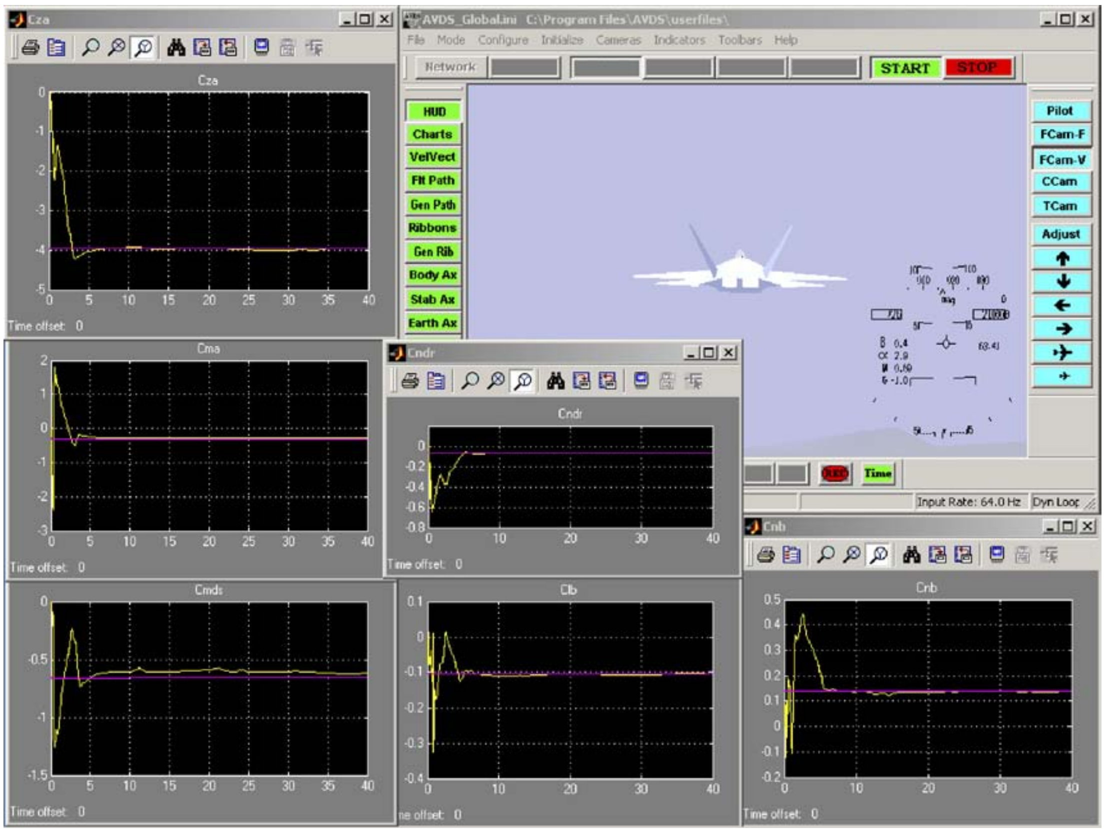
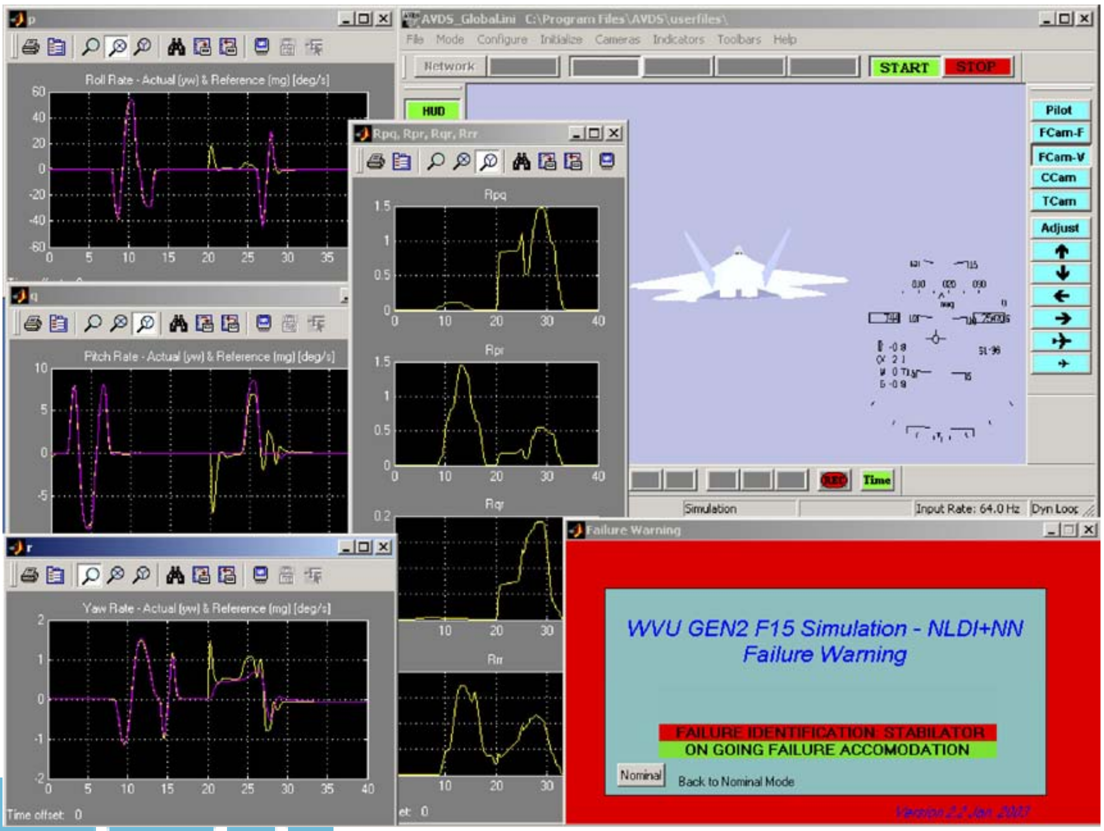


**Figure 13** WVU IFCS simulation environment graphics – direct adaptive flight control laws


p
Roll Rate - Actual (yw) & Reference (mg) [deg/s]
q
Pitch Rate - Actual (yw) & Reference (mg) [deg/s]
r
Yaw Rate - Actual (yw) & Reference (mg) [deg/s]
Time offset: 0
Rpq, Rpr, Rqr, Rrr
Rpq
Rpr
Rqr
Rrr
AVDS_Global.ini C:\Program Files\AVDS\userfiles\
File Mode Configure Initialize Cameras Indicators Toolbars Help
Network
START
STOP
HUD
Pilot
FCam-F
FCam-V
CCam
TCam
Adjust
REC
Time
Simulation
Input Rate: 64.0 Hz
Dyn Loop
Failure Warning

WVU GEN2 F15 Simulation - NLDI+NN
Failure Warning

FAILURE IDENTIFICATION: STABILATOR
ON GOING FAILURE ACCOMODATION

Nominal Back to Nominal Mode

Version 2.2 Jan 2007

parameter estimation. Direct adaptive flight control laws consist of NLDI performed at a reference nominal flight condition augmented with artificial NNs.

A detailed GUI for simulation scenario setup allows the selection of control laws architecture, components, and parameters as well as the general flight conditions including different failure scenarios.

Owing to its modularity, flexibility, and rapid reconfiguration capabilities of the major components, the simulation environment provides excellent means for fault-tolerant control system design, testing, comparison, and evaluation.

## References

Antoniewicz, R.F., Duke, E.L. and Patterson, B.P. (1988), "User's manual for interactive LINEAR, a Fortran program to derive linear aircraft models", NASA Technical Paper 2835.

Boeing (1999), *Intelligent Flight Control: Advanced Concept Program – Final Report*, BOEING-STL 99P0040, The Boeing Company, Chicago, IL.

Boskovic, J.D., Li, S.M. and Mehra, R.K. (2001), "On-line failure detection and identification (FDI) and adaptive reconfigurable control (ARC) in aerospace applications", *Proc. of the American Control Conference, Arlington, VA.*

Calise, A.J. and Sharma, M. (2000), "Adaptive autopilot design for guided munitions", *AIAA Journal of Guidance, Control and Dynamics*, Vol. 23 No. 5.

Calise, A.J., Lee, S. and Sharma, M. (1998), "Direct adaptive reconfigurable control of a tailless fighter aircraft", *Proc. of the 1998 AIAA Guidance, Navigation and Control Conference, Boston, MA*, AIAA 98-4108.

Campa, G., Fravolini, M.L., Napolitano, M.R., Perhinschi, M.G. and Battipede, M. (2004), "A stochastically optimal feedforward and feedback technique for flight control systems of high performance aircrafts", *Proceedings of the American Control Conference, Boston, MA*, pp. 1713-8.

Hageman, J.J., Smith, M.S. and Stachowiak, S. (2003), *Integration of Online Parameter Identification and Neural Network for In-flight Adaptive Control*, Technical Report NASA/TM-2003-212028, NASA Dryden Flight Research Center, Edwards, CA.

Halyo, N., Direskeneli, H. and Taylor, B. (1992), "A stochastic optimal feedforward and feedback control methodology for superagility", NASA CR 4471, November.

Kaneshige, J., Bull, J. and Totah, J.J. (2000), "Generic neural flight control and autopilot system", AIAA 2000-4281.

Krishnakumar, K., Limes, G., Gundy-Burlet, K. and Bryant, D. (2003), "An adaptive critic approach to reference model adaptation", *Proc. of the AIAA Guidance, Navigation, and Control Conference.*

Lu, Y., Sundararajan, N. and Saratchandran, P. (2000), "Analysis of minimal radial basis function network algorithm for real time identification of nonlinear dynamic systems", *IEEE Proceedings on Control Theory and Application*, Vol. 4 No. 147, p. 476.

Monaco, J.F., Ward, D.G. and Bateman, A.J.D. (2004), "A retrofit architecture for model-based adaptive flight control", *Proc. of the AIAA 1st Intelligent Systems Technical Conf., Chicago, IL.*

Morelli, E.A. (1999), "Real-time parameter estimation in the frequency domain", *Proceedings of the 1999 AIAA Atmospheric Flight Mechanics Conference*, AIAA Paper 99-4043, Portland, OR.

Napolitano, M.R., Song, Y. and Seanor, B. (2001), "On-line parameter estimation for restructurable flight control systems", *Aircraft Design*, Vol. 4.

Napolitano, M.R., Younghawn, A. and Seanor, B. (2000), "A fault tolerant flight control system for sensor and actuator failures using neural networks", *Aircraft Design*, Vol. 3 No. 2.

Nguyen, N.T., Bakhtiari-Nejad, M. and Huang, Y. (2007), "Hybrid adaptive flight control with bounded linear stability analysis", *Proc. of the AIAA Guidance, Navigation, and Control Conference and Exhibit, AIAA 2007-6422, Hilton Head, SC.*

Nguyen, N.T., Krishnakumar, K., Kaneshige, J. and Nespeca, P. (2006), "Dynamics and adaptive control for stability recovery of damaged asymmetric aircraft", *Proc. of the AIAA Guidance, Navigation, and Control Conference and Exhibit, Keystone, CO.*

Perhinschi, M.G., Napolitano, M.R., Campa, G. and Fravolini, M.L. (2004), "A simulation environment for testing and research of neurally augmented fault tolerant control laws based on non-linear dynamic inversion", *Proceedings of the AIAA Modeling and Simulation Technologies Conference, Providence Rhode Island.*

Perhinschi, M.G., Napolitano, M.R., Campa, G., Fravolini, M.L. and Seanor, B. (2007), "Integration of sensor and actuator failure detection, identification, and accommodation schemes within fault tolerant control laws", *Control and Intelligent Systems*, Vol. 35 No. 4, pp. 309-18.

Perhinschi, M.G., Campa, G., Napolitano, M.R., Lando, M., Massotti, L. and Fravolini, M.L. (2002a), "A simulation tool for on-line real time parameter identification", *Proceedings of the AIAA Modeling and Simulation Conference, Monterey, CA*, AIAA2002-4685.

Perhinschi, M.G., Campa, G., Napolitano, M.R., Lando, M., Massotti, L. and Fravolini, M.L. (2006a), "Modeling and simulation of a fault tolerant control system", *International Journal of Modelling and Simulation*, Vol. 26 No. 1, pp. 1-10.

Perhinschi, M.G., Lando, M., Massotti, L., Fravolini, M.L., Campa, G. and Napolitano, M.R. (2002b), "On-line parameter estimation issues for the NASA IFCS F-15 fault tolerant systems", *Proceedings of the American Control Conference, Anchorage, AK*, pp. 191-6.

Perhinschi, M.G., Napolitano, M.R., Campa, G., Seanor, B., Burken, J. and Larson, R. (2006b), "An adaptive threshold approach for the design of an actuator failure detection and identification scheme", *IEEE Transactions on Control Systems Technology*, Vol. 14 No. 3, pp. 519-25.

Rassmussen, S. (2000), *Aviator Visual Design Simulator (AVDS) – User Manual*, Rassmussen Simulation Technologies, Columbus, OH.

Rauw, M.O. (1998), *FDC 1.2 – A SIMULINK Toolbox for Flight Dynamics and Control Analysis*, Delft University of Technology, Delft.

Shin, Y. and Ghosh, J. (1991), "The pi-sigma network: an efficient higher-order neural network for pattern classification and function approximation", IEEE 7803-0164.

Steinberg, M.L. (1999), "A comparison of intelligent, adaptive, and nonlinear flight control laws", AIAA 99-4044, August.

US Department of Defense (1980), *Military Specification – Flying Qualities of Piloted Airplanes*, US Department of Defense, Washington, DC, MIL-F8785C.

Williams-Hayes, P.S. (2005), *Flight Test Implementation of a Second Generation Intelligent Flight Control System*, Technical Report NASA/TM-2005-213669, NASA Dryden Flight Research Center, Edwards, CA.

## Corresponding author

**M.G. Perhinschi** can be contacted at: Mario.Perhinschi@mail.wvu.edu