Inductive Learning Approaches for Improving Pilot Awareness of Aircraft Faults

ARTICLE · SEPTEMBER 2005

DOI: 10.2514/6.2005-6990

Lilly Spirkovska

NASA

39 PUBLICATIONS **301** CITATIONS

SEE PROFILE

David L. Iverson

NASA

37 PUBLICATIONS **158** CITATIONS

SEE PROFILE

Scott Poll

NASA

36 PUBLICATIONS **553** CITATIONS

SEE PROFILE

Inductive Learning Approaches for Improving Pilot Awareness of Aircraft Faults

Lilly Spirkovska, David L. Iverson, Scott Poll, and Anna Pryor
NASA Ames Research Center, Moffett Field, CA 94035-1000, USA

Neural network flight controllers are able to accommodate a variety of aircraft control surface faults without detectable degradation of aircraft handling qualities. Under some faults, however, the effective flight envelope is reduced; this can lead to unexpected behavior if a pilot performs an action that exceeds the remaining control authority of the damaged aircraft. The goal of our work is to increase the pilot's situational awareness by informing him of the type of damage and resulting reduction in flight envelope. Our methodology integrates two inductive learning systems with novel visualization techniques. One learning system, the Inductive Monitoring System (IMS), learns to detect when a simulation includes faulty controls, while two others, Inductive Classification System (INCLASS) and multiple binary decision tree system (utilizing C4.5), determine the type of fault. In off-line training using only *non-failure* data, IMS constructs a characterization of nominal flight control performance based on control signals issued by the neural net flight controller. This characterization can be used to determine the degree of control augmentation required in the pitch, roll, and yaw command channels to counteract control surface failures. This derived information is typically sufficient to distinguish between the various control surface failures and is used to train both INCLASS and C4.5. Using data from *failed* control surface flight simulations, INCLASS and C4.5 independently discover and amplify features in IMS results that can be used to differentiate each distinct control surface failure situation. In real-time flight simulations, distinguishing features learned during training are used to classify control surface failures. Knowledge about the type of failure can be used by an additional automated system to alter its approach for planning tactical and strategic maneuvers. The knowledge can also be used directly to increase the pilot's situational awareness and inform manual maneuver decisions. Our multi-modal display of this information provides speech output to issue control surface failure warnings to a lesser-used communication channel and provides graphical displays with pilot-selectable levels of details to issues additional information about the failure. We also describe a potential presentation for flight envelope reduction that can be viewed separately or integrated with an existing attitude indicator instrument. Preliminary results suggest that the inductive approach is capable of detecting that a control surface has failed and determining the type of fault. Furthermore, preliminary evaluations suggest that the interface discloses a concise summary of this information to the pilot.

I. Introduction

At the start of aviation, aircraft were relatively simple and required only a few gauges to inform the pilot of the state of the aircraft. As aircraft became more complex, more gauges were necessary and the pilot was required to track the state of many more systems. What began as a single-pilot operation increased to a three-pilot operation. One pilot is dedicated to monitoring the state of aircraft systems while the other two pilots aviate, navigate, and communicate with air traffic controllers. The introduction of automation has enabled the redesign and consolidation of the three-pilot operation into a two-pilot operation. Many of the tasks previously assigned to a human pilot are now accomplished by automation, with supervision from the human pilots. The automation can take the form of an autopilot capable of flying the aircraft nearly from take-off to landing, an automated monitoring and diagnosis system that continuously evaluates the health of the aircraft's systems, or systems that offer other assistance. Automation has many advantages over non-automated operations and, together with aeronautics advances and crew-resource management (CRM), has made aircraft flights safer. CRM stresses that it is very important for the pilots to work together as a team, that all team members need to keep each other informed of their activities, and everyone must feel comfortable questioning another member's methods. The "strong, silent type" airline captain of the past who took complete control, gave orders, and did not react well to being questioned has been displaced from today's flight deck. Unfortunately, another team member with similar characteristics has been welcomed and encouraged: automation.[4] Although automation has proven to be an effective resource that can decrease the pilots' workload, problems can arise when unexpected events occur.

Our research has two main goals. One goal is to make automation technically better – to improve its capabilities. The other goal, which addresses the concerns of re-introducing a strong, silent type into the flight deck, is to improve the pilot's awareness of what the automation is doing. The domain of our work is automated neural network flight controllers that can successfully accommodate a variety of aircraft control surface faults without detectable degradation of aircraft handling qualities. This adaptation is transparent to the pilot and does not lead to an increase in workload. However, some faults can lead to a reduction in the effective flight envelope (i.e., the area in which the aircraft responds to pilot action as expected) and ensuing unexpected behavior if the pilot performs an action that exceeds the remaining control authority of the damaged aircraft. To help prevent this occurrence, it is imperative that the pilot is informed of the existence and character of a problem and how it may affect the flight, including limitations on maneuvering, speed, landing gear deployment, and flap deployment. Ideally, if completion of the mission were in doubt, the pilot would also be informed of available alternatives. Our research focuses on identifying and informing the pilot about the type of damage. We also present a method to inform the pilot of envelope reduction in the pitch and roll axes, but present only a preliminary notion of how to compute the reduction due to the damage.

In the following sections, we will describe each of the major functions of the health management process flow shown in Fig. 1: simulator, fault detection, fault isolation/classification, envelope reduction, and user interface. Within the fault isolation section, we present a comparison of the two classification methods. The scale, filter/smooth, and buffer/summarize functions will not be described as they are only supporting details. We conclude with potential areas for future work.

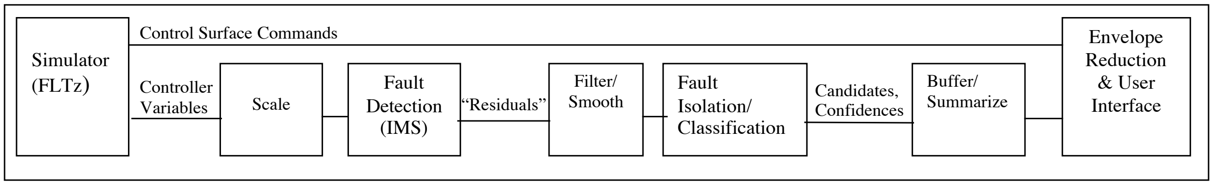


Figure 1: Health management process flow.

II. The Problem

A neural network flight controller[1] uses both pre-trained and on-line learning neural networks to provide adaptive flight control. The pre-trained neural networks provide estimates of aerodynamic stability and control characteristics required for model inversion. The on-line learning networks generate command augmentation signals to compensate for errors in the stability and control derivative estimates as well as errors from model inversion. An additional benefit of the on-line learning networks is that they can learn to adapt to changes in aircraft dynamics due to damage or failure.

The inputs to the neural network flight controller are commands generated by the pilot through rudder pedal and lateral and longitudinal control stick displacements. Rudder pedal and stick gains are used to convert displacement commands into roll rate and aerodynamic normal and lateral acceleration commands. These acceleration commands are then transformed into the corresponding roll rate, pitch rate, and yaw rate commands. Finally, the necessary control surface deflections are computed using dynamic inversion of the rate commands and reference models for first-order roll rate, pitch rate, and yaw rate transfer functions that specify the desired handling qualities. Thus, neural network flight controllers use aircraft state information to generate pseudo control augmentation commands in order to compensate for errors computed from aircraft feedback. The pilot is not explicitly aware of this process. Nor is he aware that the aircraft dynamics may have changed due to damage or failure. The neural network flight controller makes the failure essentially transparent to the pilot.

This process becomes problematic only when the controller is not able to make the failure completely transparent. Under some failures, the effective flight envelope is reduced, and if the pilot goes just outside this envelope, he is suddenly and unexpectedly faced with the reality of the failure. Even worse, he may not be able to easily determine the type of failure and therefore will not know how to react to regain control of the aircraft.

In this paper, we present two methods we developed to assist the pilot in determining the type of failure. Both methods begin by learning to detect that a failure has occurred (*fault detection*). They then independently categorize the failure (*fault isolation*). Using the results of the isolation algorithms, we can then compute how the flight envelope is reduced. We present only a notional implementation of this aspect. Finally, we suggest a method of presenting the results of the detection, isolation, and envelope reduction to the pilot.

III. Simulator

For a training and demonstration platform, we used the FLTz flight simulator.[2] It is a 6 degrees-of-freedom, non-linear, high-fidelity simulator that can simulate a variety of aircraft and controller types. Our simulations used a transport aircraft similar to the Boeing 757 that incorporates a fault tolerant, adaptive neural network flight control system that does not provide fault detection and isolation. FLTz has generic control panels and flight displays that are representative of typical aircraft displays, facilitating development, evaluation, and demonstration.

FLTz can simulate many types of failures. Our prototype was concerned only with faults affecting ailerons, horizontal stabilizer, elevators, and rudder. Moreover, we limit the type of damage to these surfaces to hard-to-position and loss-of-effectiveness control surface faults. A hard-to-position (HTP) fault results when the control surface is stuck in position, that is, the control surface moves to a fixed position but does not respond to subsequent commands. A loss-of-effectiveness (LOE) fault results from reduced gain in the mechanisms that drive the control surface elements (i.e., degraded actuation) and results in less control surface movement than expected for a given command signal. Note that 100% LOE is equivalent to an HTP at the trim position, that is, the mechanism that drives the control surface has suffered a 100% reduction in the ability to move the surface to a commanded position and thus, the surface remains in its current position.

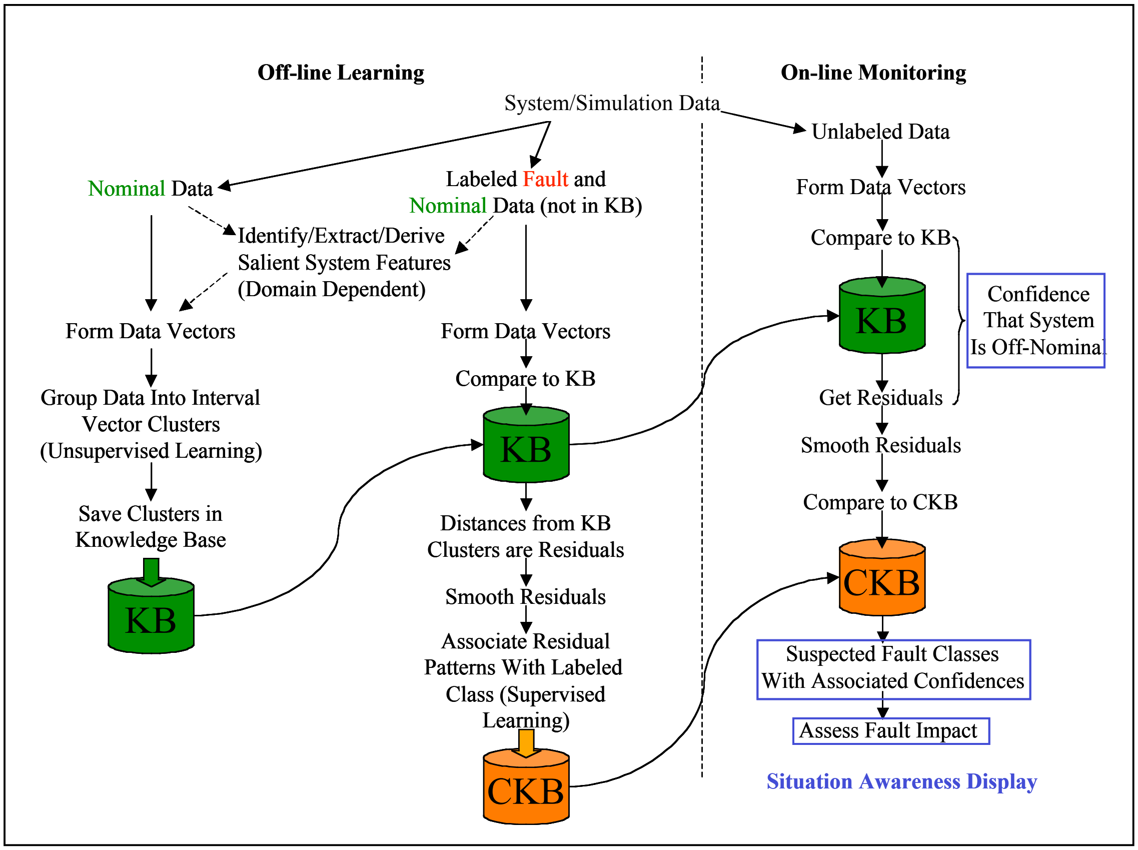

Off-line Learning
On-line Monitoring
System/Simulation Data
Nominal Data
Labeled Fault and Nominal Data (not in KB)
Identify/Extract/Derive Salient System Features (Domain Dependent)
Form Data Vectors
Group Data Into Interval Vector Clusters (Unsupervised Learning)
Save Clusters in Knowledge Base
KB
Form Data Vectors
Compare to KB
KB
Distances from KB Clusters are Residuals
Smooth Residuals
Associate Residual Patterns With Labeled Class (Supervised Learning)
CKB
Unlabeled Data
Form Data Vectors
Compare to KB
KB
Confidence That System Is Off-Nominal
Get Residuals
Smooth Residuals
Compare to CKB
CKB
Suspected Fault Classes With Associated Confidences
Assess Fault Impact
Situation Awareness Display

Figure 2: Approach of an inductive learning system. (KB = knowledge base; CKB = class KB)

The detection and isolation methods we developed are inductive and follow the approach depicted in Fig. 2. To generate training data, two pilots independently flew the simulator under a nominal cruise condition defined by mach 0.78 to 0.84, altitude between 28,000 and 36,000 feet, and turbulence from none to moderate. For each training set, the pilots flew a series of roll and pitch maneuvers: roll 10º left then right, roll 20º left then right, roll 30º left then right, pitch 5º up then down, and pitch 10º up then down. A typical training flight is shown in Fig. 3. In total, we generated 50 minutes of data, roughly 60% of which was used to train IMS.

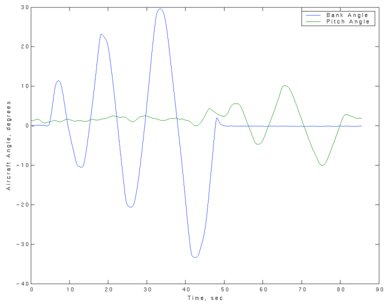


Figure 3: Example of training flight.

IV. Fault Detection

The neural flight controller compensates for control surface failures with little feedback to the pilot. Aircraft handling qualities can sometimes degrade as the neural network adapts its weights to compensate for the changes in aircraft dynamics. Even if the pilot detects this degraded handling, he is not informed of the type of failure. Thus, the pilot does not have an opportunity to prepare to take over control or avoid maneuvers that may lead to unacceptable behavior. To provide visibility into the activities of the controller, the first task is to detect that a fault exists. The Inductive Monitoring System (IMS)[5] accomplishes this task.

IMS combines features of model-based reasoning and data clustering to build knowledge bases of nominal system behavior. For off-line training, it begins with a set of data vectors that capture nominal flight control system behavior. The vectors, derived from 30 minutes of the two pilots' flights described in the previous section, consist of a description of the operating conditions (roll, pitch, and yaw commands; and the rate of change in roll, pitch, and yaw commands) and features that describe the neural flight controller's behavior (effective lateral stick position, effective longitudinal stick position, and effective pedal displacement). These nominal vectors are grouped into interval vector clusters of nearby points that specify acceptable ranges for parameters in a vector. When forming the clusters, the operating conditions are weighted more than the neural flight controller's behavior features.

During on-line monitoring, IMS uses the knowledge base of clusters computed in the off-line training to determine whether incoming data is nominal. Incoming data is captured in a vector of the same structure as the training set. The vector is then compared to the clusters in the knowledge base and the closest nominal cluster is identified. Finally, a distance is computed for each of the three feature elements based on the difference between the incoming element value and the corresponding element value of the closest cluster. These three feature distances are the residuals. The confidence level that a problem exists is based on a Gaussian function of overall feature distance. Figure 4 shows a comparison of feature data from a faulty simulation and a nominal simulation, the residual from a simulator running in parallel with the same control input, and the IMS output comparing faulty output to the nominal knowledge base. Note the IMS percent deviation is shown in the bottom graph and clearly shows its correspondence to the faulty simulation feature values.

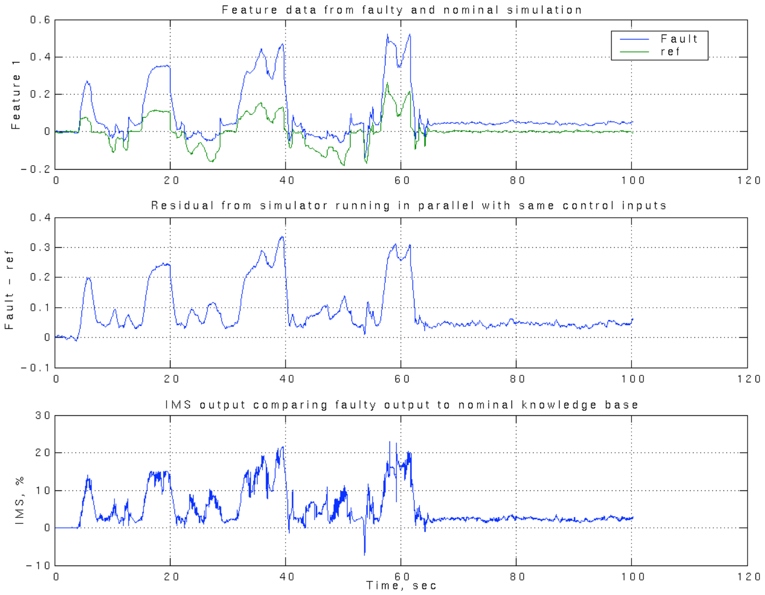


Figure 4: IMS output showing deviation from nominal of a simulator flight with an injected fault.

V. Fault Isolation

Just detecting a fault increases the pilot's situational awareness. Knowing what type of fault has been detected is also very valuable because it allows the pilot to modify his maneuvers to remain within the effective flight envelope of the damaged aircraft. We used two classification techniques to determine the type of fault. One technique, the Inductive Classification System (INCLASS), uses similar methods to IMS. The other uses a public domain decision tree classification system, C4.5.[3] Both techniques are pre-trained on approximately half the data set generated from five-minute faulty flights for each fault type (HTP or LOE for the six control surfaces [left/right aileron, left/right elevator, horizontal stabilizer, and rudder]). Note that the magnitude of HTP and LOE faults was discretized in 5º increments to reduce the training set and evaluate how well each technique generalizes to untrained fault magnitudes. We briefly describe each technique and then discuss and compare the performance of each technique in isolating the fault.

A. Inductive Classification System

The INCLASS methodology is similar to IMS. It is trained on the IMS output residuals for labeled nominal and failure data. The training set consists of 60 classes, with each fault magnitude considered a separate class. Class membership and confidence are based on the (Gaussian-weighted) relative distance of the real-time data vectors from clusters representing each class. Thus, if a data vector is within a predefined in-class threshold of N clusters, as shown in Fig. 5, INCLASS will output the fault classes represented by those N clusters as the candidates and determine the confidence level for each fault based on the distances from the input vector to the clusters.

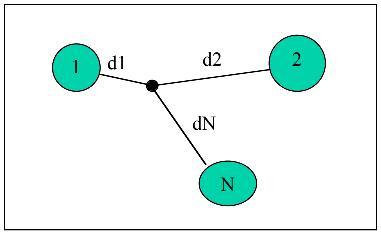


Figure 5: Distance of input vector (black disk) to N closest clusters determines membership and confidence level.

B. Multiple Binary Decision Trees

In general, decision trees are constructed based on the information gain ratio for attribute/value pairs. That is, when building the tree during off-line training, the training data set is divided into subsets by selecting an attribute whose values most differentiate one class from the others. The attribute with this characteristic becomes the root of the tree and the values that attribute can take on become branches. The same algorithm is recursively applied to each subset along each branch of the tree until the subset of data along the bottom branches is homogeneous, i.e., of a single class. In real applications, it is not always possible to create a decision tree with a homogeneous set of data at the bottom branches that is capable of classifying unknown data correctly. The decision tree must be tuned to minimize false alarms and have acceptable classification capability. One tuning method is to continue branching only until reaching an acceptable level of classification, e.g., exceeding a threshold percentage of training data that is classified correctly.

Our second technique uses multiple binary decision trees implemented in C4.5 and tuned to determine the fault type. They are multiple in the sense that a separate decision tree is built for each of twenty fault classes: +HTP (i.e., control surface stuck in up position), -HTP (surface stuck in down position), and LOE for each control surface, plus nominal (no-fault) case and neural flight controller system (NFCS) failures. And they are binary in the sense that the trained fault is labeled class 1 and all other faults as well as the nominal class are labeled class 0. For real-time classification, each binary tree is executed with the IMS residual. All trees that indicate that the given IMS residual is of class 1 (i.e., member) are reported as candidates. The confidence for each candidate class is based on the percentage of training data that is classified correctly for a particular leaf in the tree. The total confidence is then divided among the multiple candidates. The fault magnitude (e.g., 10º up HTP) is determined by a rule base and heuristics.

C. Classifier Performance

The performance of INCLASS and C4.5 on the various faults is presented in the tables of Fig. 6. As stated above, INCLASS was trained on 60 fault classes. However, for a direct comparison with C4.5, we condense the 60 classes into the 20 classes used to train C4.5. The goal, 100% accuracy, would result in a 1 along the diagonal and 0 elsewhere. C4.5 reached that goal only for rudder HTP failures. For HTP failures of the remaining surfaces, both INCLASS and C4.5 achieved at least 94% accuracy with C4.5 performing slightly better than INCLASS. The tables also show how a fault was misclassified. In most cases, HTP faults were either classified correctly or were not classified as faults (that is, they were misclassified in the *None* group although IMS detected they were somehow faulty).

The performance of both classifiers was worse for classifying LOE failures, with INCLASS outperforming C4.5 by from 3% (rudder LOE) to 26% (right elevator LOE). Similar to the classifiers' performance on HTP failures, the performance of both classifiers was best for rudder LOEs, with INCLASS achieving nearly 99% accuracy and C4.5 achieving nearly 96%. For LOE faults of other surfaces, INCLASS achieved from a low of 70% for a left elevator LOE to a high of 98% for a stabilizer LOE, and C4.5 achieved from a low of 48% for a right elevator LOE to a high of 94% for a stabilizer LOE. The 60-class INCLASS results indicate the right elevator LOE performance was only 9% for a magnitude-25% LOE, but in the mid-70s for the other three magnitudes. The misclassified data was primarily classified either as left elevator HTP –5 (48%), right aileron HTP 0 (19%), and rudder LOE magnitude-25% (19%). In fact, the first two fault types (L elev HTP –5 and HTP 0) account for the majority of the misclassifications of all LOE faults. This leads us to believe that one cause of the decreased classification rate on these faults is due to the aircraft's performance with an LOE fault. The controller reacts differently than for nominal conditions, but the classification vector does not have the features necessary to distinguish between the various small-magnitude faults. For relatively small maneuvers with small-magnitude LOE failures, the feature variable signatures are very similar to that of a nominal aircraft. Thus, the flight controller compensates in a more subtle way, one that is more difficult for the classifiers to determine. Additionally, with an HTP fault, the flight controller must compensate for the stuck control surface by utilizing the other surfaces. For most of the HTP failures, the neural flight controller is continuously counteracting the control surface that is stuck at a non-trim position even when the pilot is not maneuvering, resulting in a steady-state offset of at least one of the three flight controller features that compose the training vector. For non-100% LOE faults, on the other hand, the control surfaces move in the correct direction, but not as quickly nor as far. There is no steady-state offset for LOE failures because the controller counteracts the slower movement of the control surface only during pilot maneuvering.

Indicated Fault => Actual Fault	none	-L ail	+L ail	L ail LOE	-R ail	+R ail	R ail LOE	-L elev	+L elev	L elev LOE	-R elev	+R elev	R elev LOE	-rud	+rud	rud LOE	-stab	+stab	stab LOE	NFCS		# points
none	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	:	110779
-L ail	0.0259	0.9733	0	0.0002	0	0.0001	0.0001	0	0	0.0001	0	0	0.0001	0	0	0	0	0	0	0.0001	:	81020
+L ail	0.0582	0	0.9407	0.0009	0.0001	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	:	49894
L ail LOE	0.0515	0.0009	0.0016	0.9044	0.0005	0.0001	0.007	0.0018	0.0001	0.0062	0.0004	0	0.0214	0	0	0.0021	0.0001	0	0.0018	0.0001	:	23875
-R ail	0.0277	0.0001	0	0.0001	0.9711	0.0001	0.0001	0	0	0	0	0	0	0	0	0.0006	0	0	0	0.0001	:	80679
+R ail	0.0566	0.0001	0	0.0001	0.0001	0.9424	0.0007	0	0	0	0	0	0	0	0	0	0	0	0	0	:	49662
R ail LOE	0.0569	0.0004	0.0004	0.0038	0.0007	0.0015	0.8885	0.0003	0.0001	0.0087	0	0.001	0.0067	0.0001	0	0.0099	0	0	0.0207	0.0004	:	27595
-L elev	0.0153	0	0	0.0003	0	0	0.0005	0.9585	0	0.011	0.0011	0	0.0121	0	0	0	0.0012	0	0	0	:	63996
+L elev	0.0181	0	0	0	0	0	0	0	0.9561	0.0012	0	0.0022	0.0003	0	0	0	0	0.0005	0.0217	0	:	98563
L elev LOE	0.0026	0.0002	0.0001	0.0025	0.0001	0.0001	0.0041	0.0045	0.0008	0.7088	0.0019	0.0099	0.1219	0	0	0.0027	0.0007	0	0.1389	0.0001	:	25866
-R elev	0.0216	0	0	0.0007	0	0	0.0003	0.0013	0	0.0029	0.9626	0	0.0095	0	0	0	0.0011	0	0	0	:	62550
+R elev	0.0192	0	0	0	0	0	0	0	0.0004	0.0006	0	0.9605	0.0004	0	0	0	0	0.0001	0.0187	0	:	95409
R elev LOE	0.001	0.0001	0	0.0015	0	0	0.0004	0.0069	0.0007	0.0528	0.0038	0.0063	0.7476	0	0	0.0072	0.001	0	0.1705	0	:	24017
-rud	0.0225	0	0	0.0004	0	0	0.0002	0	0	0	0	0	0	0.9769	0	0	0	0	0	0	:	64508
+rud	0.0338	0	0	0	0	0	0	0	0	0	0	0	0	0	0.9621	0.0041	0	0	0	0	:	62720
rud LOE	0.005	0	0	0.0009	0.0002	0	0.0004	0.0005	0	0.0009	0	0	0.0025	0	0.0003	0.9893	0	0	0	0	:	77168
-stab	0.0048	0	0	0.0002	0	0	0.0005	0.0056	0	0.0155	0.0024	0	0.0185	0	0	0.0001	0.9468	0	0.0056	0	:	32962
+stab	0.0146	0	0	0	0	0	0	0	0.0016	0.0015	0	0.0023	0.0004	0	0	0	0	0.9649	0.0147	0	:	34627
stab LOE	0.0016	0	0	0	0	0	0	0	0.0006	0.0047	0	0.0052	0.0018	0	0	0	0	0	0.986	0	:	116685
NFCS	0.0073	0.0024	0.0003	0.0146	0.0014	0	0.0038	0	0	0.0049	0	0	0.0003	0	0	0.0024	0	0	0	0.9624	:	2875
Accuracy: 1133663 of 1185430 (95.63%)																						

Indicated Fault => Actual Fault	none	-L ail	+L ail	L ail LOE	-R ail	+R ail	R ail LOE	-L elev	+L elev	L elev LOE	-R elev	+R elev	R elev LOE	-rud	+rud	rud LOE	-stab	+stab	stab LOE	NFCS		# points
none	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	:	110779
-L ail	0.016	0.984	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	:	79556
+L ail	0.0255	0	0.9745	0.0000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	:	49764
L ail LOE	0.1719	0	0.0023	0.8205	0	0	0	0	0	0.0053	0	0	0	0	0	0	0	0	0	0	:	23888
-R ail	0.0235	0	0	0	0.9765	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	:	79287
+R ail	0.022	0	0	0	0	0.978	0	0	0	0	0	0	0	0	0	0	0	0	0	0	:	49553
R ail LOE	0.2037	0	0	0	0	0.0014	0.7876	0	0	0	0	0	0	0	0	0.0007	0	0	0.0067	0	:	27618
-L elev	0.0473	0	0	0	0	0	0	0.9511	0	0	0	0	0	0	0	0	0.0016	0	0	0	:	62270
+L elev	0.0214	0	0	0	0	0	0	0	0.9757	0	0	0	0	0	0	0	0	0	0.0029	0	:	95922
L elev LOE	0.449	0	0	0	0	0	0	0	0	0.5291	0	0	0	0	0	0	0	0	0.0218	0	:	25873
-R elev	0.0357	0	0	0	0	0	0	0	0	0	0.9643	0	0	0	0	0	0	0	0	0	:	60833
+R elev	0.0245	0	0	0	0	0	0	0	0	0	0	0.9732	0	0	0	0	0	0	0.0023	0	:	92768
R elev LOE	0.4661	0	0	0	0	0	0	0	0	0	0	0	0.4898	0	0	0	0.0091	0	0.0351	0	:	24019
-rud	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	:	62633
+rud	0.0022	0	0	0	0	0	0	0	0	0	0	0	0	0	0.9978	0	0	0	0	0	:	60820
rud LOE	0.0393	0	0	0	0	0	0	0	0	0	0	0	0	0	0.0018	0.959	0	0	0	0	:	77142
-stab	0.0519	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.9481	0	0	0	:	30966
+stab	0.0236	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.9721	0.0043	0	:	34204
stab LOE	0.0595	0	0	0	0	0	0	0	0.0007	0	0	0	0	0	0	0	0	0	0.9399	0	:	114935
NFCS	0.2651	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.7349	:	2878
Accuracy: 1102052 of 1165708 (94.54%)																						

Figure 6: Performance of INCLASS (top) and C4.5 (bottom) on classifying HTP (not labeled) and LOE (labeled) faults. The ±L/±R represents up/down for the left surface and up/down for the right surface for each control surface (aileron "ail", elevator "elev", stabilizer "stab," and rudder "rud"). NFCS is the neural flight controller system. See discussion in Section VII.

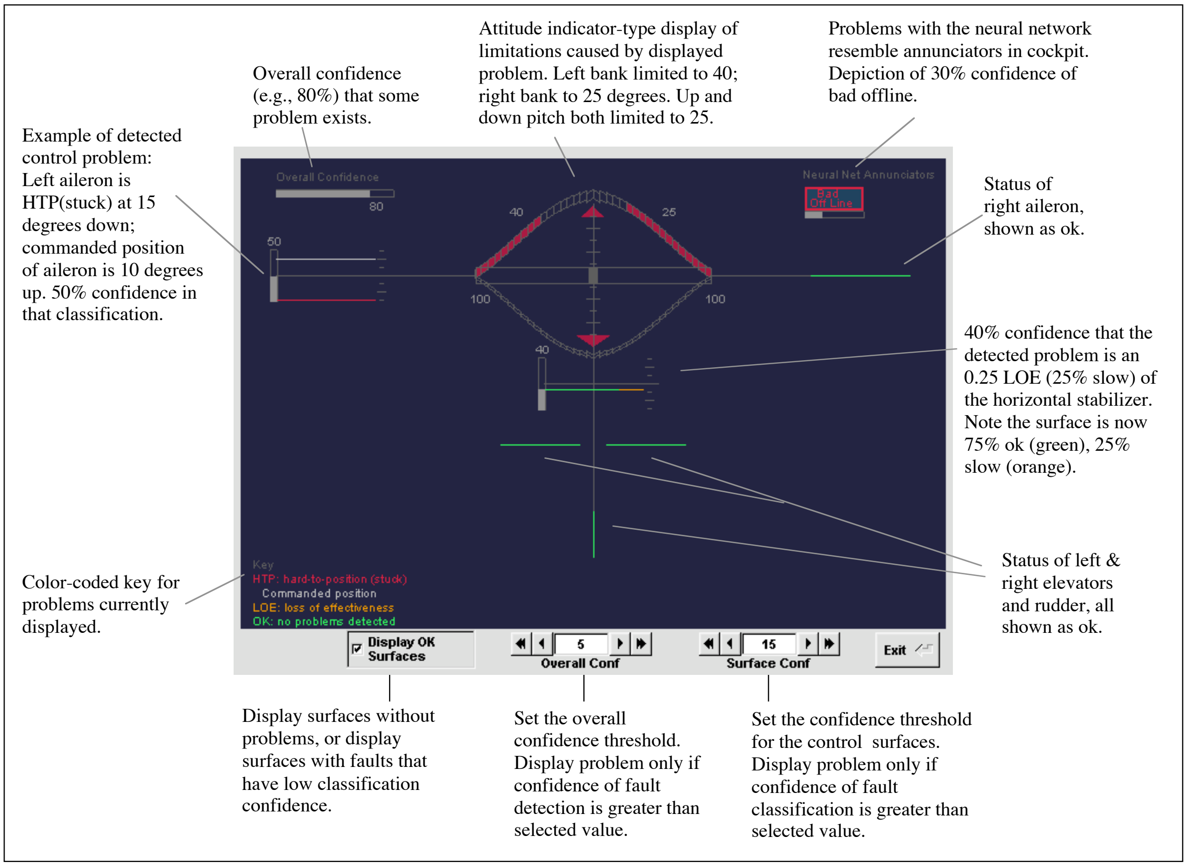
VI. Envelope Reduction

Fault detection and isolation provide necessary information to the pilot about the health state of the aircraft. In addition to knowing what components failed and how they are failed, it is important to know the failure's impact on mission success. Envelope reduction aims to provide this information.

Our current implementation of envelope reduction is only for concept demonstration and is mostly notional. A rigorous analysis[6] is necessary but is outside the scope of our work. Thus, we applied heuristics derived from a simple analysis of how a certain fault will affect maneuverability. The heuristics considers such factors as aileron failures affect roll control authority whereas elevator and stabilizer failures affect pitch control authority. We were unable to test these heuristics empirically, so we will not enumerate them here.

VII. User Interface

An important aspect of providing information to a pilot is presenting it in a way that does not increase the pilot's workload.[7,8,9] This is especially important when presenting additional visual information because of the many visual displays already present in a modern flight deck. Our method of addressing this concern is illustrated in Fig. 7. The visualization and graphical user interface for displaying fault information was influenced both by pilots and by human factors experts. The pilots, including two of the authors, provided advice on what is important to a pilot. The human factors experts provided advice on how to best encode the information based on existing human factors principles. These opinions of both groups are expressed in the discussion below and helped determine color choices.



Overall confidence (e.g., 80%) that some problem exists.

Attitude indicator-type display of limitations caused by displayed problem. Left bank limited to 40; right bank to 25 degrees. Up and down pitch both limited to 25.

Problems with the neural network resemble annunciators in cockpit. Depiction of 30% confidence of bad offline.

Example of detected control problem: Left aileron is HTP(stuck) at 15 degrees down; commanded position of aileron is 10 degrees up. 50% confidence in that classification.

Status of right aileron, shown as ok.

40% confidence that the detected problem is an 0.25 LOE (25% slow) of the horizontal stabilizer. Note the surface is now 75% ok (green), 25% slow (orange).

Color-coded key for problems currently displayed.

Status of left & right elevators and rudder, all shown as ok.

Display surfaces without problems, or display surfaces with faults that have low classification confidence.

Set the overall confidence threshold. Display problem only if confidence of fault detection is greater than selected value.

Set the confidence threshold for the control surfaces. Display problem only if confidence of fault classification is greater than selected value.

Figure 7: Graphical user interface showing faults, confidence levels, and envelope reduction.

For our application, the pilot needs to know whether there is a fault, the type of fault, the automation's confidence level both that a fault exists and that it is of the annunciated type, and the limitations caused by the fault. If a fault does not exist or the automation's confidence level of fault existence is below a selected threshold, the screen is dark. This abides by the dark-cockpit principle: maintain a blank screen unless a problem exists. This principle not only minimizes distraction to the pilots when things are normal, but also increases saliency of caution and warning signs when something goes wrong. Note that the color key in the lower-left corner of the display also follows the dark-cockpit principle and disappears under nominal conditions. The key serves as a memory aid of the color-coding scheme, not a lengthy explanation of what each fault type means. By showing a partial set of keys corresponding to the currently shown defects, we minimize display clutter and possibly increase the saliency of displayed information. For example, if an LOE fault becomes an HTP fault of the same surface, popping up a new red key simultaneously with changing the color of the control surface to red may attract additional pilot attention.

At the pilot's request (accomplished by selecting the appropriate option "Display OK Surfaces"), non-problematic control surfaces can be displayed to allow verification that everything is working as expected. Following human factors principles for color-coding for improved recognition, these surfaces are represented by green, the prototypical color meaning *nominal*. The color key is modified to show only that green represents "OK: no problems detected."

When a fault is detected, the confidence level that a fault exists is displayed in the upper-left corner, *Overall Confidence* in Fig. 7. The level is displayed numerically for precise information and graphically for quick visual interpretation. Note that the position of the number moves relative to the filled portion of the bar to coincide with the location the pilot would already be looking for one of the two pieces of information. The gray color chosen for the confidence bars, both for overall confidence and for detected faults, is lower contrast than the colors chosen for state of control surfaces and envelope reduction. This allows the pilot to see the information but does not overwhelm the other, probably more important, information.

When a classifier isolates a detected fault to a particular fault-type with a confidence level that exceed the *Surface Conf* level specified by the pilot, the fault is displayed on the airplane mimic. Figure 7 shows two faults: a stabilizer LOE of magnitude 25% and a left aileron HTP of magnitude 15º down. Continuing the color-coding scheme, LOE faults are shown as amber, meaning *caution*. The amount of amber versus the amount of green reflects the percentage of loss of effectiveness. Thus, for a 25% LOE, the surface is depicted 75% in green and 25% in amber. Similarly, a 75% LOE is depicted 25% in green and 75% in amber. At 100% LOE, which is equivalent to HTP at trim position, the color changes to red to signify the *warning* aspects of the fault. Similarly, HTP faults are depicted in red. Additionally, HTP faults display the position at which the surface is stuck and the position the surface was commanded to. The commanded position is shown in gray for similar reasons as the confidence bars. For LOE and HTP failures, the classifier's confidence level is also presented to help the pilot determine the amount of trust to place in the annunciation. Lastly, the color key is modified to show the meaning of the colors currently displayed. In the example of Fig. 7, all possible fault-types are shown and, hence, the color key is completely shown.

Another element shown in Fig. 7 is envelope reduction. The attitude-indicator inspired display in the top-middle shows the limitations for bank, pitch, and yaw. The bank limitations are shown in red, representing *warning*, on the upper perimeter of the envelope display. For example, the limits shown in Fig. 7 represent a maximum available 40º of left bank and 25º of right bank. Although those numbers were derived using simplistic heuristics, the display illustrates the limited range available to the pilot after a failure for which the neural flight controller compensation results in diminished control authority on that (roll/pitch/yaw) axis. Similarly, up and down pitch limitations due to the stabilizer failure (in this case) are shown by the red triangles. Each tick marks represents 5º; hence, the pilot is limited to no more than 25º pitch up and down. Larger limitations are shown using larger triangles that fill more of the interior of the envelope display. Thus, a large limitation in pitch would be very quickly discernable. Finally, yaw limitations, are represented by filling in the horizontal bar in the middle of the envelope display. The two "100" to either side of that bar represent that the aircraft shown in Fig. 7 has no limitations (100% availability) in yaw. If a limitation exists, the bar fills from the outside in and, similar to the pitch display, a large area of red represents a large limitation.

The final element shown, in the upper-right corner of the display in Fig. 7, is *Neural Net Annunciators*. In particular, a *Bad Off Line* is outlined in red. At this point, the classifiers are not trained to recognize neural network faults (e.g., bad off line (prior-to-flight) training of the neural net). Although we have some data on neural flight controller faults (*NFCS* in the table of Fig. 6), we do not have enough to teach the classifiers to recognize the fault consistently. Thus, the display shows only a concept for how such faults could be displayed once the information is available.

In addition to the comprehensive visual display, we implemented a simple speech interface to aurally warn the pilot of a detected fault. We used the public domain Festival speech synthesizer.* When IMS detects a fault with a high (pre-defined) confidence, Festival says, for example, "Left aileron stuck." This prompts the pilot to look at the visual display for additional details.

VIII. Conclusions

We have demonstrated a method that uses inductive learning to improve pilots' awareness of aircraft faults that are masked by an adaptive flight controller. The Inductive Monitoring System, IMS, detects the existence of a fault during real-time flight of an adaptive flight controller. Two inductive classification approaches, INCLASS and multiple binary decision trees (implemented with C4.5), isolate the fault to either a hard-to-position (HTP) fault or loss-of-effectiveness (LOE) fault of one of six control surfaces and determine the magnitude of the failure (e.g., HTP (stuck) at 10º up). Based on the detected fault, a heuristics-based approach determines the resulting limitations of the aircraft, that is, the reduction in the effective flight envelope due to the failures. Finally, the information from these three systems and the envelope reduction computation is displayed graphically for quick interpretation by the pilot.

IMS is an effective method of detecting faults. Although trained only on a small set of nominal data, it was able to detect off-nominal conditions with high accuracy. IMS was also able to determine control augmentation residuals relative to fault tree performance without the use of a compute-intensive simulation running in parallel. This capability provides automatic nominal system characterization from data without the need for dedicated simulator development. It also allows for rapid updates to system characterization (either off-line or on-line) in the event of reconfiguration, upgrades, or damage.

* http::/www.cstr.ed.ac.uk/projects/festival/

INCLASS and C4.5 were trained on a small set of off-nominal data. Both generalized very well for HTP failures but not as well for LOE failures. We believe the better generalization is due to the method of compensation for HTP failures by the neural flight controller. Whereas the controller compensates for HTP failures even when the pilot is not maneuvering, it compensates for LOE failures only during a maneuver that requires that surface. Further, the LOE fault signatures for small magnitude faults did not differ much from the nominal data signatures. Because the labels for the INCLASS training set specified not only the type of fault but also the magnitude, INCLASS performed better than C4.5 on LOE failures. Based of these differences, in future work, it may be advantageous to fuse the results of the two classifiers. The result presented to the pilot could be a weighted combination of the results from each classifier, with the weighting learned over time to reflect the performance of each classifier on a specific fault type.

In addition to fusing the classifier results, further areas for research include determining whether classification results can be improved by expanding the training set to include additional pilots, applying other classification techniques for improved recognition of faults, improving the flight envelope reduction method, investigating the detection of failures of the neural network itself rather than assuming faulty aircraft behavior is due to failures of control surfaces, expanding the methodology to other flight phases, especially approach, and determining whether one classifier can be trained to detect faults in multiple phases or if multiple classifiers are necessary.

Over and above improving the fault detection and fault isolation algorithms, an evaluation of the user interface should be conducted. The current design is based on input from pilots and human factors experts in a static environment. The pilots' requirement for information may change with extended use of the system. For example, the actual fault may be irrelevant and the only necessary information is the envelope reduction. Conversely, the confidence of the detection and isolation systems may prove very important and require further emphasis. Finally, the method of displaying the two types of failures needs to be assessed in flight to verify that the amber portion of LOE failures is visible and the two types of failures are easily and quickly distinguishable.

Acknowledgments

This work was supported by the Information Technology Strategic Research (ITSR) project of the Computing, Information and Communications Technology (CICT) program at NASA Ames Research Center. The authors thank Rob McCann and Miwa Hayashi for their assistance with the GUI design and Scott Christa and Don Bryant for their assistance with data collection.

References

1 Rysdyk, R.T. and Calise, A.J., "Fault Tolerant Flight Control via Adaptive Neural Network Augmentation," *AIAA Guidance, Navigation, and Control Conference and Exhibit,* Boston, MA, 1998, AIAA-1998-4483.

2 Kaneshige, J., Bull, J., and Totah, J., "Generic Neural Flight Control and Autopilot System," *AIAA Guidance, Navigation, and Control Conference and Exhibit,* Denver, CO, 2000, AIAA-2000-4281.

3 Quinlan, J.R., *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1993.

4 Norman, D.A., "Coffee Cups in the Cockpit," *Turn Signals are the Facial Expressions of Automobiles*, Cambridge, MA: Perseus Publishing, 1992, Chap. 16.

5 Iverson, D., "Inductive System Health Monitoring," *Proceedings of the 2004 International Conference on Artificial Intelligence (IC-AI'04)*, Las Vegas, NV, 2004, pp. 605-611

6 Yavrucuk, I., Prasad, J.V.R., Calise, A.J., and Unnikrishnan, S., "Adaptive Limit and Control Margin Prediction and Limit Avoidance," *Proceedings of the American Helicopter Society 58th Annual Forum*, Montreal, Canada, 2002.

7 Endsley, M.R., "Situation Awareness in Aviation Systems," *Human Factors in Aviation Systems*, edited by D.J. Garland, J.A. Wise, and V.D. Hopkin, Hillsdale, NJ: Lawrence Erlbaum, 1999, pp. 257-276.

8 Reeves, L.M., et al., "Guidelines for multimodal user interface design," *Communications of the ACM*, Vol. 47, No. 1, 2004, pp. 57-59.

9 Flach, J.M., "Situation Awareness: Proceed with Caution," *Human Factors*, Vol. 37, No. 1, 1995, pp. 149-157.