Airborne Electro-Mechanical Actuator Test Stand for Development of Prognostic Health Management Systems

Edward Balaban[1], Abhinav Saxena[2], Sriram Narasimhan[3], Indranil Roychoudhury[4], Kai F. Goebel[5], and Michael T. Koopmans[6]

[1] *NASA Ames Research Center, Intelligent Systems Division, MS 269-3, Moffett Field, CA 94035, USA*
edward.balaban@nasa.gov
[2,4] *SGT Inc., NASA Ames Research Center, Intelligent Systems Division, MS 269-4, Moffett Field, CA 94035, USA*
abhinav.saxena@nasa.gov
indranil.roychoudhury@nasa.gov
[3] *UARC, NASA Ames Research Center, Intelligent Systems Division, MS 269-4, Moffett Field, CA 94035, USA*
sriram.narasimhan-1@nasa.gov
[5] *NASA Ames Research Center, Intelligent Systems Division, MS 269-4, Moffett Field, CA 94035, USA*
kai.goebel@nasa.gov
[6] *Oregon State University, Corvallis, OR, 97331, USA*
koopmans@engr.orst.edu

ABSTRACT

With the advent of the next generation of aerospace systems equipped with fly-by-wire controls, electro-mechanical actuators (EMA) are quickly becoming components critical to safety of aerospace vehicles. Being relatively new to the field, however, EMA lack the knowledge base compared to what is accumulated for the more traditional actuator types, especially when it comes to fault detection and prognosis. Scarcity of health monitoring data from fielded systems and prohibitive costs of carrying out real flight tests create the need to build high-fidelity system models and design affordable yet realistic experimental setups. The objective of this work is to build an EMA test stand that, unlike current laboratory stands typically weighing in excess of one metric ton, is portable enough to be easily placed aboard a wide variety of aircraft. This stand, named the FLEA (for Flyable Electro-mechanical Actuator test stand), allows testing EMA fault detection and prognosis technologies in flight environment, thus substantially increasing their technology readiness level – all without the expense of dedicated flights, as the stand is designed to function as a non-intrusive secondary payload. No aircraft modifications are required and data can be collected during any available flight opportunity: pilot currency flights, ferry flights, or flights dedicated to other experiments. The stand is currently equipped with a prototype version of NASA Ames developed prognostic health management system with models aimed at detecting and tracking several fault types. At this point the team has completed test flights of the stand on US Air Force C-17 aircraft and US Army UH-60 helicopters and more experiments, both laboratory and airborne, are planned for the coming months.[*]

1. INTRODUCTION

As mentioned earlier, EMA are being groomed to become the workhorse of the next generation fly-by-wire aircraft and spacecraft. The advantages of these devices over the traditional hydraulic controls are well understood – lower recurrent maintenance, lower overall weight (no need for heavy hydraulic reservoirs and lines), better survivability in combat situations and superior suitability for operation in harsh environments. What is not yet well characterized, however, are the EMA fault modes and the nature of their progression under varying operational and environmental conditions.

The motivation for this effort came from the realization of just how difficult it is to obtain high quality EMA performance data (under nominal conditions and even more so in the presence of a fault) suitable for prognostic work. Manufacturers, understandably, in most cases consider this type of information to be proprietary. EMA deployed in the field (in military or civilian aircraft, for example) are often not instrumented sufficiently to provide useful data. While there have been some past efforts to collect

EMA performance data in flight (Jensen et al, 2000), they were done with nominal actuators and for relatively short periods of time.

The question often arises as to why go through the trouble of actually putting an actuator test stand on an airplane? Why can't the same type of data be collected solely on the ground, in laboratory conditions? Our reasons for pursuing this path are twofold:

- Some of the environmental conditions, such as background vibration, acoustic noise, and G-loads are difficult to reproduce in the laboratory environment. Without testing the entire system in relevant conditions, one cannot be certain that, for example, a particularly clever feature extracted from accelerometer signals in laboratory environment is going to be as useful in an environment with a high external level of vibration
- With PHM-augmented EMA driving a primary control surface of an aircraft being the ultimate demonstration, the FLEA is a necessary intermediate step needed to demonstrate technology maturity (for instance, the ability to operate in a real-time mode, among other benchmarks). Using an unobtrusive, self-contained system like the FLEA, this can be done in a relatively fast, inexpensive, and safe manner.

The design and fabrication of the test stand started at California Polytechnic State University under the guidance from the Diagnostic and Prognostic Group of the NASA Ames Research Center. After a fast-paced development effort, resulting in a mechanically and electrically complete system, the stand was transferred to NASA Ames Research Center, where the development of the control, data acquisition, user interface, and data processing software was completed. Other work included expansion of the sensor suite and data acquisition capabilities, hardening of all the systems for flight environment, and developing components necessary for communication with various types of aircraft.

The remainder of the paper will expand further on the design considerations that guided the team in creating the FLEA, describe the hardware and the software components in Section 2, discuss the design philosophy of the experiments in Section 3, and go into details of the current health management system in Section 4. Some ideas on future directions of this research will be outlined in Section 5.

2. THE FLEA

The key idea of this work is to design, build and fly a self-contained, lightweight test fixture containing three actuators: one nominal, one injected with faults, and the third providing dynamic load. The load is switched in-flight from the healthy to the faulty test actuator, thus providing the fault injection capability for the test stand without having to modify the actuator in flight. The stand is connected to the aircraft data bus and the motion profiles for the test actuators, as well as the load applied to them, are derived from the corresponding real-time values for one of the aircraft's control surfaces.

Some of the main constraints in designing the FLEA were size, weight, and cost. It was desirable to make the stand compact enough to be able to fit into a standard 19-inch equipment rack present on some aircraft. A compact size in general also allows placement aboard a wider variety of aircraft. The same reason motivated the desire to keep the weight of the FLEA to a minimum, so that it can potentially even be flown aboard some of NASA's unmanned aerial vehicles. The cost was minimized by utilizing off-the-shelf and in-house built components as much as possible.

The FLEA is a largely self-contained unit. The only external interfaces required are those for the aircraft data bus and power. Power is provided via 110V AC and 28V DC connectors. The 110V AC line is used to power the processing and data acquisition unit via a 340W power supply and the 28V DC line is used to power the actuators and some of the sensors.

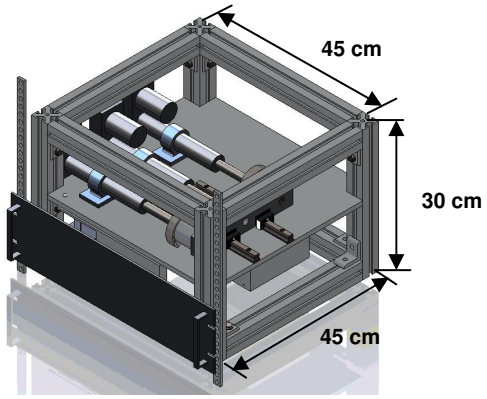

Figure 1. FLEA test stand engineering model

2.1 Hardware

Chassis

The frame is constructed from T-slotted extruded aluminum segments connected with brackets and fasteners. This allows for a relatively quick disassembly for servicing between experiments. The 1 cm thick center plate is attached to the frame and used for mounting the actuators and other components of the stand. Rigidity of the central plate was an important design consideration, therefore analysis was performed that showed only negligible bending under the expected

loads. Before a flight, the sides of the chassis (except for the top) are covered with 3 mm thick aluminum plates. These plates serve a dual purpose: as an additional safety measure in case of a crash and to provide EMI protection. The top portion of the stand, where EMI emissions are not of a concern, is protected with a thick panel of high-strength Lexan™, which allows the operator to visually observe the test in progress. Structural analysis demonstrated that no internal components would be able to pierce either the aluminum or the Lexan panels during a crash up to 20g in deceleration. EMI protection provided by the shielding satisfies MIL-STD-461.

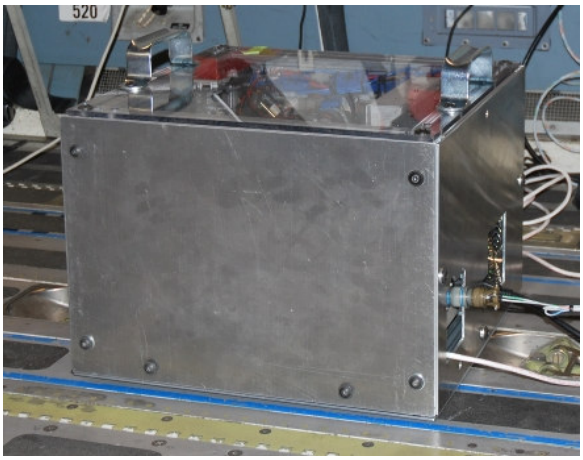

Figure 2. FLEA aboard an aircraft with protective panels mounted

The overall weight of the FLEA, with all the components mounted, is approximately 35 kg. The stand is typically mounted either in the aforementioned instrumentation rack or strapped/bolted to the floor of the fuselage.

Processing Unit

The processing unit, running the operating system, data acquisition, control, and health management software, is based on an off-the-shelf Pentium 4 3.2GHz ATX form-factor motherboard. A standard set of input/output ports is supported (RJ-45, USB, PCI, ePCI, and RS-232).

Storage is provided by two solid-state drives. A smaller, 32 GB one is used for the software, while a larger, 128GB one, is dedicated to data storage. Solid-state drives were chosen over traditional hard drives for their ability to operate at high altitudes without the need for pressurization.

Sensor Suite and Data Acquisition System

The data acquisition system consists of two NI 6259 cards and supports a comprehensive sensor suite, which is described in Table 1.

Table 1. FLEA Sensor Suite

Sensor	Qty	Type	Location
Load cell	1	Omega LC703-75	Between the load actuator and the test actuator
Accelerometer	2	Endevco 7253C	On the nut of the ball screw
Thermocouple	4	T type	On the ball screw nut and motor housing
Rotary encoder	2	UltraMotion E5DIFF optical encoder with differential output	On the test actuator motors
Linear potentiometer	1	UltraMotion precision linear potentiometer	Along the load actuator screw
Voltage Sensor	3	Custom	Motor controller boards
Current sensor	3	Custom	Motor Controller Boards

The accelerometers are connected through custom fabricated conditioner boards that supply them with excitation voltage and remove the DC portion of the return signal. The voltage and current sensors are implemented via voltage dividers on the motor controller boards.

Control System

The test actuators are controlled via a Polulu VNH3SP30 controller. The load actuator is driven via a custom controller created at CalPoly. Coupling of test actuators to the load actuator is accomplished via electro-magnets (shown on Figure 3), with coupling commands coming through the NI 6259 cards. Only one test actuator at a time is normally coupled to the load actuator.

Test Actuators

The test articles used in the FLEA at present are UltraMotion Bug actuators. While architecturally equivalent to the larger (and considerably more

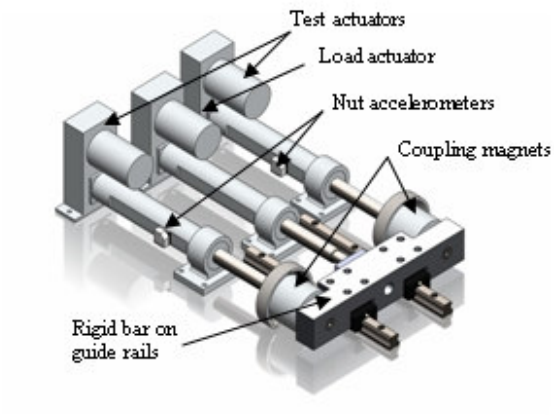

Figure 3. FLEA actuator coupling system

expensive flight-qualified EMAs), these 'off the shelf' units allow the team to conduct run-to-failure experiments in a cost-effective manner. Their high-level specifications are presented in Table 2.

Table 2. UltraMotion Bug Actuator Specifications

Mechanism type	Ballscrew with a DC electric motor
Screw thread pitch	0.125 in/rev
Efficiency	98%
Dynamic load	5000 lb*in/sec (100% duty cycle)
Motor stall torque	41.3 oz/in
Motor no-load speed	102.5 rev/sec
Motor stall current:	8.11 amps
Motor no-load current:	0.16 amps

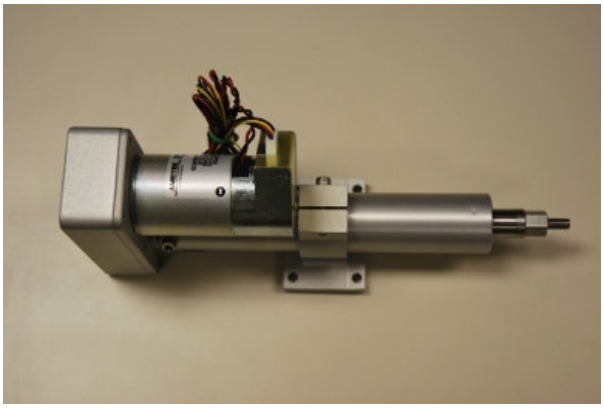

Figure 4. UltraMotion Bug actuator

2.2 System Software

The system software consists of an actuator control system, data acquisition system, flight interface system, a diagnostic system and a prognostic system. All of these are implemented in LabVIEW™ on a Windows XP operating system. However the underlying algorithms for the diagnoser and prognoser are implemented in MATLAB™. The LabVIEW code has been implemented in modular fashion with modules for different functions and data sharing through synchronization structures like queues. The overall control architecture is illustrated on the following figure:

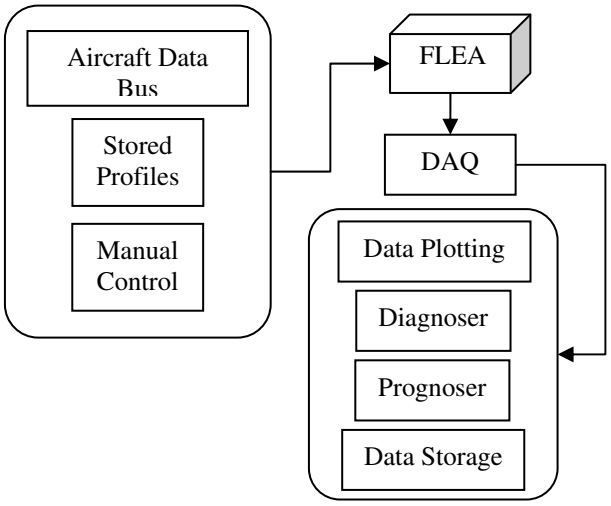

Figure 5. FLEA control architecture

The details of each of the subsystems follow next.

Actuator Control System

As described in the previous section, the actuator motors are controlled by a microcontroller board. This microcontroller is responsible for controlling the duty cycle of each actuator motor driver and returning their position. The microcontroller is controlled using serial communication over USB. LabVIEW VIs are used to send serial commands to control all three actuators as well as to determine their positions.

One of the more typical operations for a test actuator is to control it to a specific position. Hence we implemented LabVIEW code which implements a PID controller that uses a set point and feedback from the position sensor to maintain the position of the test actuator. The load actuator, as the name suggests, is intended to maintain a desired load profile and hence a similar PID controller was implemented that uses a load set point and feedback from the load cell.

Since the two test actuators are currently meant to be used one at a time, the actuator control also implements code to activate the electro-magnets,

couple the actuators, re-couple the actuators (if they get de-coupled) and switch actuators when an experiment needs to be directed to the other test actuator. The front panel for the actuator control system that provides the entire range of control actions is illustrated in Figure 6. Examples of display windows, available with the manual control interface (as well as with other types of interfaces described further in the paper) are shown on Figure 7.

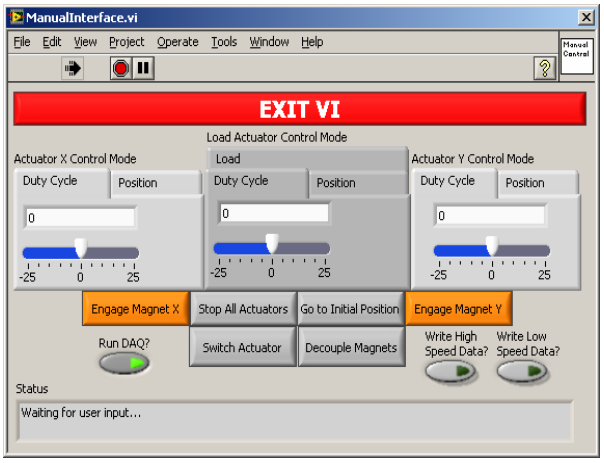

Figure 6. Manual control interface

Data Acquisition System

The data acquisition is performed via two National Instruments 6259 cards. This allows for acquiring data from some of the sensors at low speed and others, like accelerometer channels, at high speed. The data acquisition is controlled by two corresponding LabVIEW DAQ tasks. The low speed data is acquired at 1 KHz sampling rate and the high speed data is acquired at 20 KHz sampling rate. Both tasks are run continuously in a while loop gathering 100 milliseconds of data on each run.

Since the actuator positions can only be obtained through calls to the microcontroller, the DAQ software also includes code to read actuator positions once every 100 milliseconds to be synchronized with rest of the data acquisition. In addition we also record other information, like PID controller outputs and desired position and load profiles for the purposes of analysis.

The acquired data is used for a variety of purposes, including plotting for visual inspection, saving to file for future analysis, and sending to diagnoser and prognoser for online reasoning. In order to achieve the above objectives in an efficient manner, a producer-consumer architecture is implemented using a LabVIEW feature called “notifier”. Every time a 100 milliseconds’ worth of data have been acquired, it is published by sending a notification. Other modules needing this data can wait for the notification (which comes with attached data) and then wake up when the notification arrives and process the data as needed.

Figure 7. Data display windows

Diagnoser

The diagnoser is responsible for monitoring the sensor data and determining whether any faults are present in the system. The details of the diagnostic algorithm, which is implemented in MATLAB, are described in Section 4. The LabVIEW interface is designed to send the acquired data to the diagnoser and print the diagnostic results received from the diagnoser. The interface to the diagnoser/prognoser is illustrated in Figure 8.

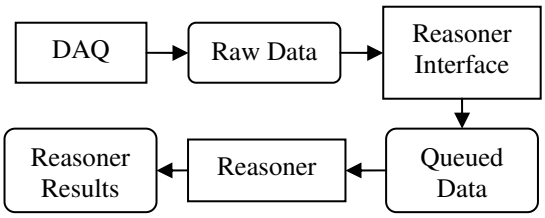

Figure 8. FLEA reasoner (diagnoser or prognoser) data pathway

Matlab Script nodes are used to interact with the MATLAB code. First an initialize function is used to set up the diagnoser. Then the acquired data is awaited using a wait-for-notification module. When data is available, it is sent to the diagnoser which reasons upon it and returns information indicating if a fault has been detected and, if so, which fault is likely to have occurred. The software goes back to waiting for more data and each time data is available the process is repeated.

Prognoser

After the diagnoser has determined that a fault has occurred in the system, the prognoser is responsible for

monitoring the sensor data and determining how the fault is progressing and how long of a useful life is remaining for the system. The details of the prognostic algorithm (which, like the diagnoser, is also implemented in MATLAB) are described later in Section 4. The LabVIEW interface is designed to send the acquired data to the prognoser and display the remaining useful life estimates received from it. This interface is also illustrated in Figure 8.

The prognostic algorithm is based on extracting features on a time-window of data points and estimating how these features are changing over time. It then determines when these features would reach the point where the system can be considered to have reached a failed state. In order to support this, the LabVIEW interface queues the acquired data and packages it into chunks to be sent to the prognoser.

Flight Interfaces

The FLEA is flown and expected to be flown on different types of aircraft, including the C-17 airlifters and UH-60 helicopters. Data from the operation of select actuators on these aircraft is fed to the FLEA control system, which exercises the test stand in an appropriately scaled manner. In order to enable this this, interfaces were created that allow data to be streamed from the data bus of each currently supported aircraft type. More interfaces will be added when the FLEA is prepared to fly on other aircraft types.

The C-17 flight interface uses a UDP packet parser to get data from the bus and then uses a specific set of conversion equations (depending on the type of C-17 actuator being mimicked) to convert the data into position and load information for the test and load actuators, respectively. The UH-60 interface reads data from a serial port and parses it based on a pre-defined UH-60 data format, then also computes the load and position profile information.

File Profile Interface

In order to test the actuators in the FLEA under different conditions, the file profile interface has been created which allows the creation and execution of specific position (for test actuators) and load (for load actuator) profiles, as well specification of fault-injection times. The data collected during the execution of such profiles on the FLEA is used to create, verify and test PHM algorithms. The file profile interface includes a profile creator/editor, a single-profile runner and a batch-profile runner. The latter executes a set of profiles separated by a predefined wait time between any two of them.

3. DESIGN OF EXPERIMENTS

The experiments were designed in such a way as to have a longer laboratory period of test article aging and fault progression monitoring followed by data collection on the same article in flight. This allows to not only to ensure a sufficient number of data points for the purposes of the prognostic system, but also to observe both the aging process and the performance of our algorithms in contrasting environmental conditions. The faults currently injected into our test actuators are described in the following section.

3.1 Injected Faults

Jam

Jam is injected by modifying the return channel of the ball screw mechanism with a small set screw that can be advanced into the channel, thus creating partial or complete blockage of it. This prevents balls from circulating and rotating properly, 'converting' the ball screw from a highly efficient rolling friction mechanism to something closer to a regular lead screw with a lower efficiency sliding friction. The jam cannot be injected while the actuator is in motion, but is otherwise easy to initiate.

Spalls

The term 'spall' refers to development of indentations in metal surfaces at high stress contact points. A severe case of a spall may result in metal flakes separating from the surface, creating potentially dangerous debris. In the case of a ball screw, where the contact surfaces of the nut and the screw (as well as the balls) may be subject to spalling, one of the consequences may be increased vibration, which can lead to damage of other actuator components. The likelihood of an EMA developing a spall in one of its components over its lifetime is not insignificant.

In the first set of experiments spalls were injected by machining a small "seed" imperfection onto the surface of the screw. In subsequent experiments the team switched to using a more precise electro-static discharge injection method. Figure 9 illustrates the locations on the screw where spalls were initiated. In order to evaluate how the size of the initial spall affects the nature of its growth, cuts of three different widths are introduced – 0.3, 0.4, and 0.5 mm (with the diameter of the bearing balls being roughly 1 mm). The depth of all cuts is kept constant at 0.3 mm. The cuts are made along the wall of the thread, extending from the top of it to the bottom of the curvature. The cuts are alternated between the left and the right sides of the thread, to ensure that fault progression in half of the fault locations occurs during either direction of motion.

At the time of this paper the data sets collected for nominal, jammed, and spalled actuators are primarily used in tuning the sensor suite and aid in designing diagnostic and prognostic features. They are also invaluable in planning for the next phase of extended duration experiments.

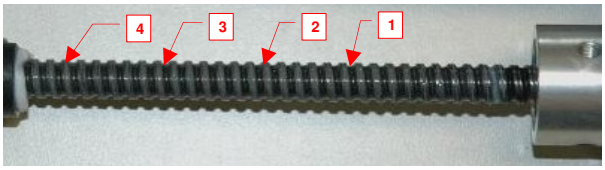

Figure 9. Spall injection

3.2 Data Collection

As briefly mentioned earlier, the data is collected in two sets of files – one for low speed measurements (current, load, voltage, position, and temperature) and another for high speed accelerometer data. The files and the data within them are time-stamped and allow for an accurate correlation with the flight log after the airborne portion of an experiment. In addition to the "follow" mode data collection, when the FLEA mimics movements and loads of an aircraft actuator, predefined profiles with set load levels and sinusoidal, triangular, or trapezoidal position trajectories are also executed both in laboratory and flight environments. This is done in order to more precisely isolate and characterize the influence of environmental conditions.

4. HEALTH MANAGEMENT SYSTEM DEVELOPMENT

The health management system being developed consists of two essential parts – the diagnostic system, tasked with detecting and identifying fault modes, and the prognostic system, that, once a fault or faults are detected, tracks their progression and provides estimates of the remaining useful life (RUL). Both of the components described below are at present considered to be proof-of-concept algorithms and more advanced versions, possibly based on different principles, are expected to be developed in the future.

4.1 Diagnostic System

Diagnostic approaches can be broadly divided into two types: *model-based* and *data-driven* (Gertler, 1998). Model-based methods rely on a system model built from *a priori* knowledge about the system, while data-driven schemes do not require such models, instead requiring large sets of exemplar failure data, which are often not available. Some of the sensors outputs available in the FLEA (such as current and voltage), can be modeled using physics-based differential equations, and then used for model-based observer-based diagnosis of faults. Modeling of accelerometer outputs, however, can be substantially more complicated, and, hence, a data-driven, feature-based diagnosis approach is better suited for leveraging accelerometer information for fault disambiguation. The FLEA diagnostic system synergistically combines model-based and data-driven diagnosis techniques in order to improve upon the either approach implemented individually.

The Transcend diagnosis approach (Mosterman and Biswas, 1999) is implemented as the model-based part of the diagnostic system (
Figure 10 illustrates its architecture). The observer, implemented as a particle filter (Arulampalam, et. al., 2002), tracks the system dynamics and estimates the unobservable system states based on the input, the control signals, and sensor measurements. In our approach, the quantitative model needed by the observer is a system of first order differential state-space equations systematically derived from the system bond model (Karnopp, et. al., 2000). Bond graphs are domain-independent, lumped-parameter models that capture energy exchange mechanisms in physical processes. The nodes of a bond graph represent energy storage, dissipation, transformation, and input-output elements, as well as two connection elements or junctions: 0- and 1-junctions (which represent parallel and series connections in the electrical domain). The connecting edges, or bonds, define energy pathways between elements.

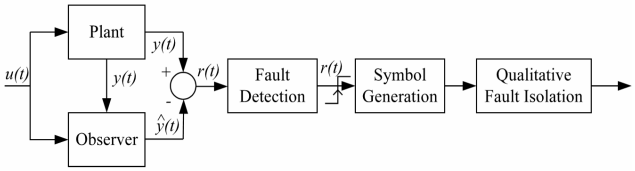

Figure 10. TRANSCEND diagnostic architecture

For *fault detection*, a statistical Z-test (Mosterman and Biswas, 1999) is used on each sensor output to determine whether the deviation of a sensor output from its nominal, expected value is statistically significant, taking into account sensor noise and other uncertainties. Once a significant deviation is detected in any one measurement, the *symbol generation* module is initiated, and *every* measurement residual, $r(t) = y(t) - \hat{y}(t)$, where $y(t)$ is the observed measurement, and $\hat{y}(t)$ is the measurement estimate calculated based on the state estimates obtained from the observer, is converted into qualitative +, -, and 0 symbols, based on whether or not the observed measurement is above, below, or at its expected nominal value.

Once a fault is detected, the *qualitative fault isolation* module determines the *fault hypotheses,* i.e., all possible system parameters and their direction of change that could explain the observed measurement deviation from nominal. To generate the fault hypotheses, a qualitative diagnostic model called the *Temporal Causal Graph*, or TCG (Mosterman and Biswas, 1999), is used, which, essentially, is a signal flow graph whose nodes represent system variables, edges denote causality information, and edge-labels denote how one variable affects another (both immediately and over time). Fault hypotheses are generated by propagating the first observed deviation backwards through the TCG.

Once identified for each fault hypothesis, the direction of change is propagated forward along the TCG to generate *fault signatures*, i.e., an ordered set of two symbols - one for magnitude, and the other for slope - which represent how each measurement residual would deviate if that fault was the only fault in the system (in this work the discussion is restricted to single faults). Once the fault signatures are generated, qualitative diagnosis involves comparing an observed deviation with the expected fault signatures of each fault for that measurement, and removing any fault hypothesis that does not explain the observed deviation from consideration, thereby refining the fault hypotheses set. Ideally it would be desirable for the qualitative module to reduce the fault hypotheses set to a singleton set. But this is not always possible in real world engineering systems. For example, in the FLEA, the spall and jam faults have similar effects on non-accelerometer sensors. Hence, qualitative model-based approach alone is not sufficient. In the described methodology, model-based component is combined with a data-driven component to further disambiguate faults and obtain better diagnostic results.

The first step for the data-driven component is to determine the features of interest to be computed from the accelerometer data. In this work, standard deviation of readings from different accelerometers is utilized for the feature set. Then, in consultation with domain experts, for each fault in the pre-defined fault set the effect on the feature were established. Note that for the purpose of this system, a single symbol determining whether or not the feature increases or decreases from nominal is sufficient to compute fault signatures for accelerometer features. Hence, a fault signature for the data-driven module is represented by a single symbol, as opposed to the two-symbol fault signature in the model-based diagnostic component.

Once the fault signature of each fault (and for each of the features) is determined, every time the qualitative diagnosis module refines its fault hypotheses set, a decision tree based on the remaining fault hypotheses is generated. The tree provides the sequence of features to be computed in order to disambiguate the maximum number of fault hypotheses in the fastest manner possible.

The model-based and data-driven modules run in parallel. When the qualitative diagnoser refines its fault hypotheses, so does its data-driven counterpart (and vice versa). As explained above, the fault signatures for features derived from expert knowledge are used for generating the next best sequence of features needed for effective disambiguation.. These features can be used to further refine fault hypotheses using an artificial neural network (Sorsa & Koivo, 1998). By combining model-based and data-driven approaches, the number of distinct features needed for disambiguation of the fault set is kept to a minimum.

4.2 The Prognostic System

A data-driven algorithm based on Gaussian Process Regression (GPR) was chosen for the initial implementation of the prognostic system. This choice was primarily governed by two factors. First, a computationally inexpensive algorithm was preferred for the initial flight phase while the entire system is being tuned for performance in a real-time environment. Second, GPR does not require explicit fault propagation models but only a heuristic rule regarding fault growth expectation. The observed data is then used to extrapolate the trend.

A Gaussian Process (GP) is a collection of random variables any finite number of which has a joint Gaussian distribution. A real GP f(x) is fully specified by its mean function m(x) and co-variance function k(x,x') defined as:

$$m(x) = E[f(x)], \tag{1}$$

$$k(x,x') = E[(f(x) - m(x))(f(x') - m(x'))] \tag{2}$$

$$f(x) \sim GP(m(x), k(x,x')) \tag{3}$$

The index set $X \in \Re$ is the set of possible inputs, which need not necessarily be a time vector. Given prior information about the GP and a set of training points $\{(x_i, f_i) \mid i = 1,...,n\}$, the posterior distribution over functions is derived by imposing a restriction on a prior joint distribution to contain only those functions that agree with the observed data points (Rasmussen and Williams, 2006). These functions can be assumed to be noisy, since in real world situations we only have access to noisy observations rather than exact function values (i.e. $y_i = f(x) + \varepsilon$, where ε is additive IID $N(0,\sigma_n^2)$). Once we have a posterior distribution, it can be used to assess predictive values for the test data

points. The following equations describe the predictive distribution for GPR (Williams and Rasmussen, 1996):

Prior

$$\begin{bmatrix} y \\ f_{test} \end{bmatrix} \sim N\left(0, \begin{bmatrix} K(X,X)+\sigma_n^2 & K(X,X_{test}) \\ K(X_{test},X) & K(X_{test},X_{test}) \end{bmatrix}\right) \quad (4)$$

Posterior

$$f_{test} \mid X, y, X_{test} \sim N\left(\bar{f}_{test}, cov(f_{test})\right), \quad (5)$$

where

$$f_{test} \equiv E[f_{test} \mid X, y, X_{test}] = K(X, X_{test})\left[K(X,X)+\sigma_n^2 I\right]^{-1} y,$$

$$cov(f_{test}) = K(X_{test}, X_{test}) - \left[K(X_{test}, X)+\sigma_n^2 I\right]^{-1} K(X, X_{test}).$$

A crucial ingredient in a Gaussian process predictor is the covariance function, $K(X,X')$, that encodes the assumptions about the functions to be learnt by defining the relationship between data points. GPR requires prior knowledge about the form of covariance function, which must be derived from the context (if possible). Furthermore, covariance functions consist of various hyper-parameters that define their properties. Setting the right values of such hyper-parameters is yet another challenge in learning the desired functions. Although the choice of a covariance function must be specified by the user, corresponding hyper-parameters can be learned from the training data using a gradient-based optimizer - such as maximizing the marginal likelihood of the observed data with respect to hyper-parameters (Mardia and Marshall, 1984).

Now that the fundamentals of GPR are described, the methodology followed for the initial phase of FLEA tests can be outlined. The tests are expected to serve a dual role for the overall research agenda. First, they will facilitate demonstration of an integrated diagnostic and prognostic system in a real-time environment. Second, the tests will result in data collected in flight operating conditions that will be extremely valuable in developing algorithms robust against environmental noise. However, these two disparate goals impose some non-overlapping requirements on the data-collection methodology, which is described in the following paragraphs.

It is expected that once the diagnostic system confirms the onset of a fault mode, the prognostic system is immediately triggered. Data collected henceforth is processed in real-time by computing relevant features. These features are then fed to the GPR algorithm for a certain period to train the algorithm parameters. The longer is the training period, the better are the chances for the algorithm to learn the true fault growth characteristics. However, a balance must be struck between the length of the training period and the risk of missing out on a sufficient prediction horizon. After the training is complete, the algorithm starts predicting trajectories of fault growth. Based on pre-specified failure threshold levels, predicted End of Life (EoL) is determined by where these trajectories cross the thresholds. Estimated EoL values can then be specified in relative terms by computing the Remaining Useful Life (RUL) values, if needed. As time passes by, more data is collected. The GPR model is updated with new observations and, subsequently, the predictions are updated. It must be noted that GPR runs into scalability issues if a long data history is utilized, as its computational complexity is $O(n^3)$. This can be alleviated using a variety of techniques. For instance, a 'forgetting factor' may be introduced to disregard old data as more recent data becomes available or a down-sampled input may be used for older data points.

In addition to demonstrating in-flight diagnostic and prognostic capabilities, an important aspect of the FLEA experiments is to be able to collect suitable data for further development of PHM algorithms between the flights. This data is required to have certain characteristics that will make them suitable for such purposes. For instance, prognostic algorithms require availability of run-to-failure data. It is quite challenging to obtain continuous run-to-failure data during a limited number of relatively short duration flights, as most of the real faults take longer than that to grow and reach the desired failure levels. Furthermore, a slow fault growth characteristic leads to a very large amount of data (multiple sensor readings combined with a high sampling frequency) with a relatively small information gain from the prognostics point of view. To alleviate this problem, several approaches are currently being tested. For instance, one could store several snapshots of high sampling rate data during a flight at predetermined intervals. This way, under the assumption of slow fault growth, we do not expect to lose important trend information with a smaller amount of data. It must be noted that this approach is not used for the diagnostic system, which requires continuous data at a lower sampling rate (and, possibly, for a smaller set of measurements). Once a fault is confirmed, a trigger is used to activate the prognostic algorithm and the higher sampling frequency data collection.

4.3 Feature Extraction

Feature extraction is a significant step in diagnostic and prognostic algorithm development. Real operating environments make it challenging to extract relevant features from data due to sensor noise in and variations in operating conditions. In the FLEA experiments two sets of features are used. The first is used for the diagnostic system. Since these features need to be

extracted on a continuous basis, they are kept relatively simple and computationally inexpensive. However, for a more targeted and detailed information extraction needed for prognosis, feature extraction methods with greater sophistication are employed. While the feature extraction process should ideally be guided by offline data analysis (if such data was already available from the plant), in the initial phase we use our prior experience with similar data sets collected in lab environments in order to narrow down the relevant feature set. Furthermore, the fault and nominal models developed for the EMAs help guide the feature extraction process. Features from the accelerometer data, such as various statistical moments, kurtosis, and spectral energies are some of the candidates most commonly used in analysis of rotating machinery. Indirect features may be developed using other measurements, such as those for current, voltage, temperature, position, and load (Balaban *et.al.,* 2009a). The final selection of the feature set for the FLEA PHM system will be made after the initial data collection and analysis phase is complete and the noise levels and peculiarities of operating conditions are well understood.

5. LABORATORY EXPERIMENTS

The team conducted the first complete round of laboratory experiments, where several types of faults were injected into test actuators and various motion and load profiles were executed by each of them. Corresponding data on nominal actuators was recorded as well. Motion profiles included periodic sinusoidal, triangular, trapezoidal and sine sweep trajectories, while the load profiles featured constant compressive and tensile forces of 10, 40, and 70 lbs, as well as zero-load scenarios. The nominal actuator experiments were conducted first, followed by jammed and spalled actuator experiments conducted using an identical set of motion and load profiles. Experiments with injected motor, load sensor, and position sensor failures were run next. More sensor faults, such as "dead sensor", bias, drift, and scaling were introduced in software for various sensors by post-processing the nominal data sets.

At the time of writing, the diagnoser is being tested on the above data, with feature extraction algorithms and fault detectors being tuned. Results so far are encouraging and will be described in detail in subsequent publications.

Long duration jam effects progression experiments are also being conducted, where two test actuators with jammed return channels are used. Figure 11 and Figure 12 illustrate one of the symptoms of a ball screw jam – increase in current consumption. Figure 11 demonstrates this for a single experimental run ("sine14") using real-time current consumption, whereas Figure 12 shows it for average current values over the entire set of experiments.

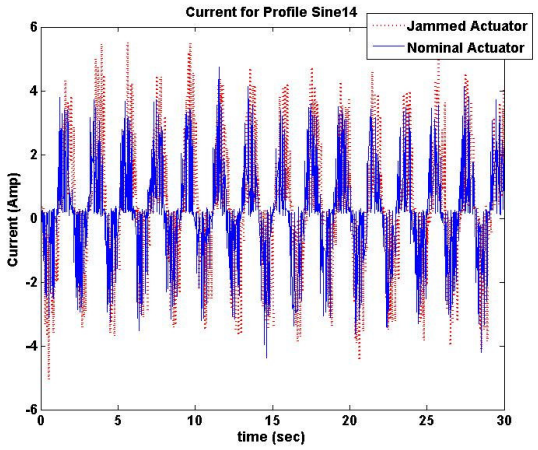

Figure 11. Current for nominal vs. jammed actuator

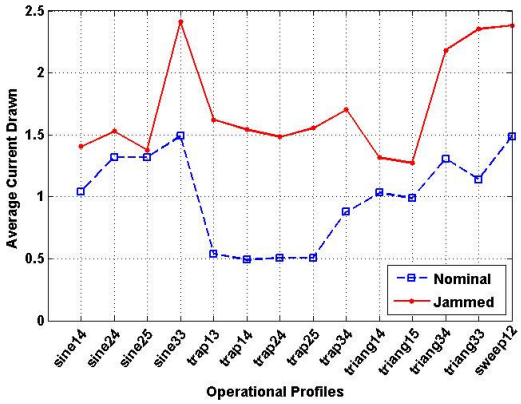

Figure 12. Current for nominal vs. jammed actuator - average values for a set of experiments

Work on analyzing fault propagation trends on the jammed actuator is continuing, with the first results expected soon. Aging experiments on the spalled actuators are slated to begin in the immediate future as well.

6. FLIGHT EXPERIMENTS

There have been several FLEA experiments on aircraft to-date. Flights on C-17 aircraft served to mature FLEA's hardware and software, while subsequent experiments on UH-60 helicopters, conducted at NASA Ames, provided valuable data sets on nominal and spall-injected actuators. This section offers a brief overview of the data collected on the UH-60, with a more extensive description and analysis to be done in a separate publication.

During the experiments, the test stand executed rigorous motion sequences, matching those of the target

UH-60 actuator (forward primary servo, an actuator responsible for pitch control of the main rotor blades). Load profiles executed by the FLEA's load actuator were derived using flight conditions information (obtained from the aircraft data bus), as well as some of the models developed by NASA's Subsonic Rotary Wing Project. The graphs below are presented for illustration purposes only, to provide the reader with some insight into the nature of the data set. Figure 13 shows a typical motion profile executed over a period of about twenty minutes. Figure 14 shows the desired (computed) load profile.

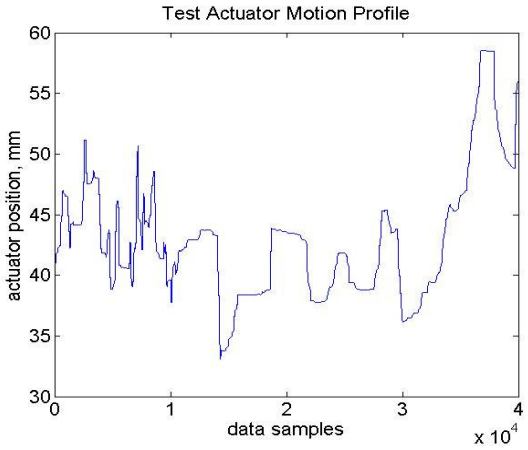

Figure 13. Test actuator motion profile during a UH-60 flight segment

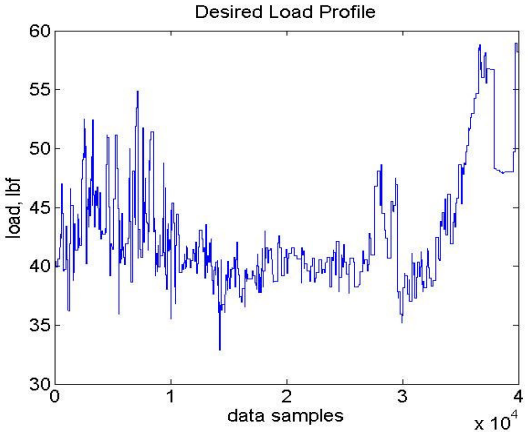

Figure 14. Desired load profile during a UH-60 flight segment

Spall fault was injected during the flight by switching the load path from the nominal to the faulty actuator. All the usual sensor readings were recorded.

7. FUTURE WORK

The team has extensive plans for the FLEA going forward. New faults, such as winding shorts and backlash, will be added to the experiments. Laboratory aging runs will be combined with further flight experiments on UH-60 helicopters and other aircraft to collect data on fault progression. The team is planning to continue improving the accuracy of the diagnostic and prognostic models and test several new algorithms. Another direction of work is making the FLEA capable of flying in an autonomous, unsupervised mode, where researchers will be able to specify ahead of time the desired time or condition of fault injection. There are also plans to expand the fault coverage from the actuator itself to the control and power components, eventually transforming the FLEA in a multi-subsystem IVHM technology demonstration.

8. CONCLUSION

While improvements to the FLEA are no doubt to continue, it already constitutes a capable platform for airborne and laboratory aging, diagnostic, and prognostic experiments. It has the capability to support a wide variety of injected fault conditions and features a capable sensor suite and data acquisition system. The FLEA can be quickly adapted to fly aboard a variety of aircraft and will soon be capable of autonomous operations. Data collection flights can be performed relatively inexpensively in a "piggy-back" mode, alongside other experiments or during pilot proficiency flights. The first test flights performed this year provided the team with useful insights on the ways to improve the test stand further and allowed to collect initial data sets to be used for feature design and next phase experiment planning (which will consist of both laboratory and airborne segments). The diagnostic and prognostic algorithms developed by the team are also being tested and refined using the new data sets. Finally, in the future the team hopes to open the FLEA to experiments conducted by research partners from other NASA organizations, industry, and the academia.

ACKNOWLEDGMENT

The authors would like to acknowledge contributions of their colleagues and collaborators at NASA Ames Research Center, NASA Dryden Research Center, California Polytechnic State University, Oregon State University, US Air Force, and US Army. The funding for this work was provided by the NASA Aviation Safety Program, IVHM Project.

NOMENCLATURE

PHM	Prognostic Health Management
FLEA	Flyable Electro-mechanical Actuator testbed
GP	Gaussian Process
GPR	Gaussian Process Regression
EMA	Electro-Mechanical Actuator

EMI	Electro-Magnetic Interference
IVHM	Integrated Vehicle Health Management
RUL	Remaining Useful Life
$y(t)$	observed measurement
$\hat{y}(t)$	measurement estimate
$r(t)$	residual
$f(x)$	Gaussian process (GP)
$m(x)$	mean function for GP
$k(x,x')$	covariance function for GP
y	noisy observations from the system
x	set of training points
ε	additive IID Gaussian noise with $N(0,\sigma_n^2)$

REFERENCES

Arulampalam, M. S., Maskell, S., Gordon, N. and Clapp, T., (2002). *A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking*, IEEE Transactions on Signal Processing, vol. 50, no. 2, pp.174-188

Balaban, E., Saxena, A, .Bansal, P., Goebel, K.F., Stoelting, P, Curran, S., (2009). *A Diagnostic Approach for Electro-Mechanical Actuators in Aerospace Systems*, IEEE Aerospace Conference 2009, Big Sky MT, pp. 1-13.

Balaban, E., Saxena, A., Goebel, K., Byington, C., Watson, M., Bharadwaj, S., Smith, M., Amin, S., (2009). *Experimental Data Collection and Modeling for Nominal and Fault Conditions on Electro-Mechanical Actuators*, Annual conference of the PHM Society, PHM09, San Diego

Balaban, E.; Saxena, A.; Bansal, P.; Goebel, K. F.; Curran, S., (2009). *Modeling, Detection, and Disambiguation of Sensor Faults for Aerospace Applications*, Sensors Journal, IEEE, vol.9, no.12, pp.1907-1917

Gertler. J. J., (1998), *Fault Detection and Diagnosis in Engineering Systems*, New York, NY: Marcel Dekker, Inc.

Jensen, S. C., Jenney, G. D., and Dawson, D., (2000), *Flight Test Experience with an Electromechanical Actuator on the F-18 Systems Research Aircraf*t, IEEE Digital Avionics Systems Conference

Karnopp, D. C., Margolis, D. L., and Rosenberg, R. C., (2000), *Systems Dynamics: Modeling and Simulation of Mechatronic Systems*, 3rd ed. NewYork, NY, USA: John Wiley & Sons, Inc.

Mardia, K. V., Marshall, R. J., (1984), *Maximum Likelihood Estimation for Models of Residual Covariance in Spatial Regression*, Biometrika, vol 71(1), pp. 135-146

Mosterman, P. J. and Biswas, G., (1999), *Diagnosis of continuous valued systems in transient operating regions*, IEEE Transactions on Systems, Man and Cybernetics, Part A, vol. 29, no. 6, pp. 554-565

Rasmussen, C. E. and Williams, C. K. I., (2006), *Gaussian Processes for Machine Learning*, The MIT Press

Sorsa, T. and Koivo, H., (1998), *Application of artificial neural networks in process fault diagnosis*, Automatica, 29(4), pp. 843–849

Smith, M.J. Byington, C.S. Watson, M.J. Bharadwaj, S. Swerdon, G. Goebel, K. Balaban, E., (2009), *Experimental and Analytical Development of Health Management for Electro-Mechanical Actuators*, IEEE Aerospace Conference, Big Sky, MT

Swerdon, G., Watson, M.J., Bhardwaj, S., Byington, C.S., Smith, M., Goebel, K., Balaban, E., (2009), *A Systems Engineering Approach to Electro-Mechanical Actuator Diagnostic and Prognostic Development*, MFPT 2009, Society for Machinery Failure Prevention Technology, Dayton, OH

Williams, C. K. I. and Rasmussen, C. E., (1996), *Gaussian Processes for Regression*, Touretzky, D. S., Mozer, M. C., and Hasselmo, M. E. (eds.), Advances in Neural Information Processing Systems, vol. 8, pp. 514-520, The MIT Press, Cambridge, MA

Edward Balaban is a researcher in the Diagnosis and System Health group at NASA Ames Research Center. He received the Bachelor degree in Computer Science from The George Washington University in 1996 and the Master degree in Electrical Engineering from Cornell University in 1997. His main areas of interest are diagnostics and prognostics of physical systems. He is currently the lead for actuator prognostics with the Diagnostics & Prognostics Group in the Intelligent Systems Division. During his years at Ames he participated in research and development of diagnostic and other autonomy elements for the X-34 experimental reusable launch vehicle, International Space Station, robotic astronaut assistants, autonomous planetary drills, and the next generation of autonomous micro-spacecraft. He is a member of IEEE.

Abhinav Saxena is a Research Scientist with SGT Inc. at the Prognostics Center of Excellence NASA Ames Research Center, Moffett Field, CA. His research involves developing prognostic algorithms and methodologies to standardize prognostics that include performance evaluation and requirement specification for prognostics of engineering systems. He is a PhD in Electrical and Computer Engineering from Georgia Institute of Technology, Atlanta. He earned his B.Tech in 2001 from Indian Institute of Technology (IIT) Delhi, and Masters Degree in 2003 from Georgia Tech.

Abhinav has been a GM manufacturing scholar and is also a member of IEEE, AIAA, AAAI and ASME.

Sriram Narasimhan is a Computer Scientist with University of California, Santa Cruz working as a contractor at NASA Ames Research Center in the Discovery and Systems Health area. His research interests are in model-based diagnosis with a focus on hybrid and stochastic systems. He is the technical lead for the Hybrid Diagnosis Engine (HyDE) project. He received his M.S and Ph.D. in Electrical Engineering and Computer Science from Vanderbilt University. He also has a M.S in Economics from Birla Institute of Technology and Science.

Indranil Roychoudhury received the B.E. (Hons.) degree in Electrical and Electronics Engineering from Birla Institute of Technology and Science, Pilani, Rajasthan, India in 2004, and the M.S. and Ph.D. degrees in Computer Science from Vanderbilt University, Nashville, Tennessee, USA, in 2006 and 2009, respectively. From September 2004 to July 2009, he has been a Graduate Research Assistant with the Institute for Software Integrated Systems, Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, Tennessee, USA. During the summers of 2006 and 2007, he was an intern with Mission Critical Technologies, Inc., at NASA Ames Research Center. Since August 2009, he has been with SGT, Inc., at NASA Ames Research Center as a Computer Scientist, Post Doctorate. His research interests include hybrid systems modeling, model-based diagnosis, distributed diagnosis, and Bayesian diagnosis of complex physical systems. He is a member of IEEE.

Kai Goebel received the degree of Diplom-Ingenieur from the Technische Universität München, Germany in 1990. He received the M.S. and Ph.D. from the University of California at Berkeley in 1993 and 1996, respectively. Dr. Goebel is a senior scientist at NASA Ames Research Center where he leads the Diagnostics & Prognostics groups in the Intelligent Systems division. In addition, he directs the Prognostics Center of Excellence and he is the Associate Principal Investigator for Prognostics for NASA's Integrated Vehicle Health Management Program. He worked at General Electric's Corporate Research Center in Niskayuna, NY from 1997 to 2006 as a senior research scientist. He has carried out applied research in the areas of artificial intelligence, soft computing, and information fusion. His research interest lies in advancing these techniques for real time monitoring, diagnostics, and prognostics. He holds ten patents and has published more than 100 papers in the area of systems health management.

Michael Koopmans is a Masters student and graduate research assistant at Oregon State University. He earned a Bachelor of Science from California Polytechnic State University San Luis Obispo in Mechanical Engineering in 2009. He has held multiple summer internships at Aperio Technologies, Solar Turbines, and most recently NASA Ames Research Center where he helped prepare flight experiments for electro-mechanical actuator fault injections. His research interests include function-based complex system design, actuator prognostics, and integration of system health algorithms. He is an ASME student member and CA certified Engineer in Training.