

# Motion-Based Thruster Fault Detection and Isolation

Edward Wilson<sup>\*</sup>, David W. Sutter<sup>†</sup>  
*Intellization, Redwood Shores, California, 94065*

Robert W. Mah<sup>‡</sup>  
*NASA Ames Research Center, Moffett Field, California, 94303*

An automatic thruster fault detection and isolation (FDI) system is a key autonomy component for thruster-controlled spacecraft. Sensor-rich FDI systems often use additional sensors such as pressure and temperature sensors in the thruster nozzles; however when building and operating smaller, less expensive spacecraft, there is a call for intelligent fault tolerance that does not increase hardware complexity, cost, and mass. A maximum-likelihood-based approach to thruster FDI is presented that can detect and isolate hard, abrupt, single- and multiple-jet on- and off-faults using only existing navigation sensors such as gyros or accelerometers. Therefore, this fault-tolerant capability could be provided as a software-only addition to a new spacecraft or modification to an existing one. These algorithms can be implemented on-board using the spacecraft's existing processor(s), on-board using a stand-alone processor, or on the ground, processing sensor information communicated to the ground stations from the spacecraft. This FDI system was originally developed through application to two specific thruster-controlled spacecraft then under development at NASA Johnson Space Center: the X-38 v.201 experimental spacecraft and the Mini AERCam. Faults are detected within one second and identified within one to five seconds in most cases for the X-38. In extended testing for the X-38, faults were correctly identified in 99.9994% of the test cases. The algorithm has since been streamlined, optimized for implementation, and extensively tested in air-bearing experiments using the MIT SPHERES experimental spacecraft. The SPHERES are scheduled for launch to the International Space Station on STS-121, with a series of experiments to provide space-flight validation following soon after.

## Nomenclature

- $\hat{a}$  = [6-by-1 vector] measured or estimated vehicle acceleration [angular ( $\hat{\alpha}$ ) and translational ( $\hat{x}^{body}$ )]
- $a_{known-system}$  = [6-by-1 vector] known-system acceleration - the acceleration that should result if no faults are present and all physical parameters are at their known values (nominal or identified, but not necessarily true)
- $\hat{a}_{disturbing}$  = [6-by-1 vector] estimated disturbing acceleration,  $\hat{a}_{disturbing} = \hat{a} - a_{known-system}$
- $\hat{a}_{disturbing,active,i}$ ,  $\hat{a}_{disturbing,inactive,i}$  = [6-by-1 vector] estimated disturbing acceleration for a sample during which fault mode  $i$  was *active* (*inactive*)
- $a_{disturbing,i}$  = [6-by-1 vector] disturbing acceleration vector corresponding to fault mode,  $i$ , based on nominal values.
- $a_{disturbing,*}$  = [6-by-1 vector] disturbing acceleration vector corresponding to the *true* fault mode
- active* – describes individual fault modes at each control update. An “off” (“on”) fault is said to be *active* if the corresponding thruster is (is not) commanded to fire during that sample period.

---

<sup>\*</sup> President, Ed.Wilson@intellization.com, AIAA Senior Member

<sup>†</sup> Senior Research Scientist, Dave.Sutter@intellization.com

<sup>‡</sup> Research Scientist, SSRL Group Lead, Robert.W.Mah@nasa.gov

- $B_k$  = [scalar] blowdown multiplier, a single scaling factor applied to all thrusters representing the reduction in thrust due to tank depletion
- $C$  = [3-by-1 vector] x-y-z coordinates of the CM in the structural frame
- $D$  = [3-by- $N$  matrix] unit vectors indicating the direction of thrust in the body frame
- $f, f_k, f_k^{body}$  = [3-by-1 vector] net force on the vehicle through the center of mass.  $k$  subscript indicates the value at time step  $k$ . *body* superscript indicates measurement in the body frame, otherwise measurement is in the inertial frame.
- $F_k$  = [ $N$ -by-1 vector] force from each thruster at time step  $k$
- $F_{bias}$  = [ $N$ -by- $N$  diagonal matrix] constant off-nominal strength of each thruster at full tank pressure
- $F_{random,k}$  = [ $N$ -by- $N$  diagonal matrix] pulse-to-pulse off-nominal strength of each thruster at full tank pressure
- $F_{nom}$  = [ $N$ -by- $N$  diagonal matrix] nominal strength of each thruster at full tank pressure
- $F_{nom,k}$  = [ $N$ -by-1 vector] nominal force from each individual thruster at time step  $k$ , accounting for estimated blowdown and firing commands
- $g()$  = Function that calculates the multi-jet multiplier,  $S_k$ , based on the number of thrusters firing simultaneously.  $S_k = g(\sum T_k)$
- $h$  = [3-by-1 vector] vehicle angular momentum
- $i$  = [scalar] fault mode number
- inactive* = opposite of *active*
- $I$  = [3-by-3 matrix] true spacecraft inertia tensor (also, dyadic, matrix), measured about the true center of mass.
- $I_{nom}$  = [3-by-3 matrix] nominal spacecraft inertia tensor, measured about the nominal center of mass
- $J$  = [3-by- $N$  matrix] x-y-z location of each thruster in the structural frame
- $k$  = [scalar] control (and ID) update counter
- $k_{window}$  = [vector] the  $k$  values for a window of measurements
- $L, L_{nom}$  = [3-by- $N$  matrix] x-y-z location of each thruster in the body frame (this changes as the [true for  $L$  and nominal for  $L_{nom}$ ] center of mass changes)
- $m$  = [scalar] vehicle total mass.
- $M_{thrusters,k}$  = [3-by-1 vector] moment on the vehicle about the CM due to the thrusters firing at time step  $k$
- $n$  = [scalar] number of degrees of freedom used in FDI
- $N$  = [scalar] number of thrusters
- $N_{window}$  = [scalar] number of sample vectors contained in the window of measurements
- $R$  = [3-by-3] direction cosine matrix used to rotate a vector from the body frame to the inertial frame (*inertial* =  $R$  *body*)
- $S_k$  = [scalar] thruster magnitude scale factor applied to all thrusters. Includes effects of blowdown and the reduction in thrust when multiple thrusters are fired.
- $T_{command,k}$  = [ $N$ -by-1 vector] which thrusters are commanded to fire at time step  $k$ . 0 when thruster is commanded off, 1.0 when commanded to fire, in between if pulse-width modulated, indicating the fraction of on time.
- $T_{failed\ off,k}$  = [ $N$ -by-1 vector of 1's and 0's] which thrusters have failed off at time step  $k$
- $T_{failed\ on,k}$  = [ $N$ -by-1 vector of 1's and 0's] which thrusters have failed on at time step  $k$

- $T_k$  = [  $N$  -by- $1$  vector of 1's and 0's] effective value for which thrusters fire at time step  $k$ , accounting for transient effects, but not for the multi-jet firing effect.
- $v()$  = function that calculates  $T_k$  as a function of present and past values of  $T_{command,k}$  as well as the true fault mode.  $T_k = v(T_{command,1}, \dots, T_{command,k}, T_{failed\ off,k}, T_{failed\ on,k})$
- $x$  = [3-by- $1$  vector] vehicle position, measured in an inertial reference frame
- $\ddot{x}, \ddot{x}^{body}$  = [3-by- $1$  vector] vehicle translational acceleration, measured in an inertial reference frame (no superscript) or the body frame (*body* superscript)
- $\hat{x}^{body}, \ddot{x}_{known-system}^{body}, \hat{x}_{disturbing}^{body}, \hat{x}_{disturbing,i}^{body}, \hat{x}_{disturbing,*}^{body}$  = [3-by- $1$  vector] translational acceleration components of  $\hat{a}, a_{known-system}, \hat{a}_{disturbing}, a_{disturbing,i}, a_{disturbing,*}$ , as defined above
- $\hat{\alpha}, \alpha_{known-system}, \hat{\alpha}_{disturbing}, \alpha_{disturbing,i}, \alpha_{disturbing,*}$  = [3-by- $1$  vector] angular acceleration components of  $\hat{a}, a_{known-system}, \hat{a}_{disturbing}, a_{disturbing,i}, a_{disturbing,*}$ , as defined above
- $\lambda_{active,i}$  = [scalar] likelihood argument for fault mode  $i$ , based on times when the fault mode is *active*
- $\lambda_{active,i0}$  = [scalar] likelihood argument for fault mode  $i$ , based on times when the fault mode is *active*, but calculated using  $\theta$  instead of  $a_{disturbing,i}$  as the expected acceleration
- $\lambda_{inactive,i}$  = [scalar] likelihood argument for fault mode  $i$ , based on times when the fault mode is not *active*
- $\lambda_{ratio,i}$  = [scalar] generalized likelihood ratio argument for fault mode  $i$ ,  $\lambda_{ratio,i} \triangleq \frac{\lambda_{active,i}}{\lambda_{active,i0}}$
- $\tau$  = [3-by- $1$  vector] sum of all torques on the vehicle, measured about the center of mass
- $\tau_{disturb}$  = [3-by- $1$  vector] sum of all torques on the vehicle resulting from other sources (drag, gravity gradient, etc.)
- $\omega$  = [3-by- $1$  vector] the angular velocity of the body-fixed frame with respect to an inertial reference frame

Superscripts:

*nom* = the variable is calculated using the nominal parameters.

*known-system* = the variable is calculated using known parameters (either nominal or identified, but not necessarily true).

*body* = measurement is in the vehicle body frame.

$\hat{\phantom{x}}$  = an estimated or identified quantity.

Coordinate frames:

Body frame – axes are aligned (parallel with, and pointing in the same direction) with the “Structural frame,” with the origin at the vehicle true center of mass.

Inertial frame – fixed in inertial space, so Newton’s laws of motion can be used without coordinate transformation.

Structural frame – fixed in the body (for example, at the geometric center).

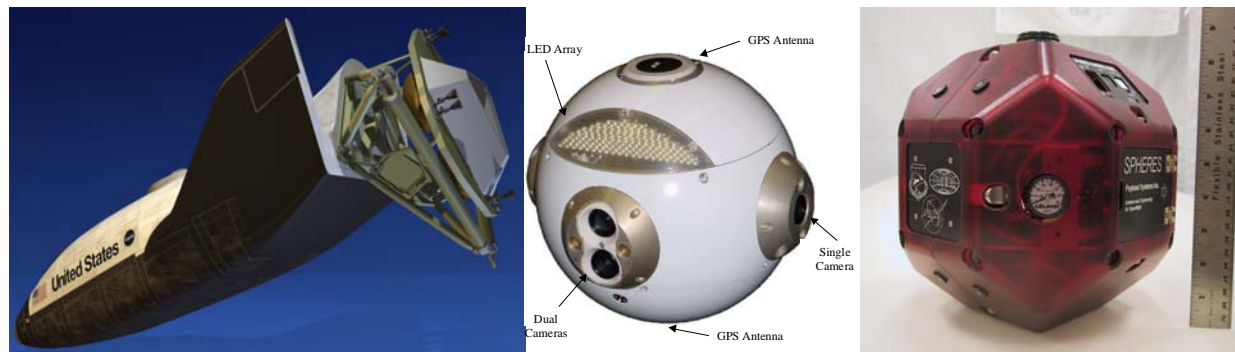
Abbreviations are listed in Appendix A.

## I. Introduction

While human-rated spacecraft are always likely to use special-purpose sensors, such as thruster-nozzle pressure sensors, for thruster fault detection and isolation (FDI), there will always also be a class of spacecraft that cannot afford the added mass, complexity, volume, and expense of these special purpose sensors. This paper presents algorithms and implementations of motion-based thruster FDI using only existing navigational sensors, such as gyros and optionally accelerometers.

This paper will present recent advances in algorithms developed to achieve motion-based thruster FDI. Initial algorithms as developed for the X-38 v.201 spacecraft and Mini AERCam are updated and extended here<sup>1</sup>. A streamlined version has been implemented on the MIT SPHERES spacecraft for upcoming experiments aboard the ISS. The above-mentioned spacecraft are shown in Figure 1.

Robust model-based FDI is always dependent to some extent on the accuracy of the model. In a closely related research effort, the authors developed algorithms for on-line identification of mass and thruster properties for thruster-controlled spacecraft<sup>2,3</sup>.



**Figure 1. Spacecraft used in development: (left to right) X-38 v.201, Mini AERCam, MIT SPHERES**

## A. Related research

Several FDI approaches reported in the literature perform well on a variety of applications<sup>4</sup>. However, the on-off nature of the thrusters present in the class of applications addressed here limits the viability of many general-purpose methods. For example, if a thruster has failed off, it will appear to be working correctly at all times that it is not commanded to fire. This report presents a general approach for this class of problems that has been validated through application to specific, realistic spacecraft applications.

This approach is especially valuable when certain application characteristics are present:

- When the faults that may appear are discrete and finite in number
- Faults occur and appear abruptly and are not intermittent
- The effect of the faults is intermittently present (active) or absent (inactive) at known times
- The measurements of these effects are imprecise due to the complexity of the governing physics and presence of sensor noise and disturbances.

The solution approach falls under the classification of model-based diagnosis, in which models of the system in its nominal and (multiple) failed conditions are used to generate predictions of the system state variables or sensor outputs. Calculation and analysis of the deviations of the measurements from predicted values is performed to detect and isolate the correct fault mode from the pre-determined, finite list of possible fault modes.

One common approach to detect and isolate spacecraft thruster faults is to install pressure, temperature, and electrical sensors at the thrusters. The use of these additional sensors makes the FDI logic very simple and robust, since they can more directly detect when a thruster is producing thrust. However, the need for additional sensors adds to the cost, complexity, mass, and volume requirements of the spacecraft. This type of system is used, for example, on NASA's Space Shuttle Orbiter. When such extensive sensing is not possible, most systems do not have automatic on-line thruster FDI capability.

Deyst and Deckert at MIT/Draper Lab developed a maximum-likelihood based approach for detecting leaking thrusters for the Space Shuttle orbiter's RCS jets<sup>5</sup>. The method for detecting soft faults was also extended to detect hard RCS jet faults. The maximum-likelihood method presented in that work is used and extended in this research.

Wilson and Rock at Stanford University developed an FDI method based on exponentially weighted recursive least squares estimation using accelerometer and angular rate sensors<sup>6,7</sup>. A neural network then provided adaptive control reconfiguration to multiple destabilizing hard and soft thruster faults. This was implemented on a 3-degree-of-freedom air-bearing vehicle.

The two preceding approaches were implemented and tested in simulation on the X-38 application, and neither one provided the required accuracy, due to the challenging signal-to-noise ratio present on the X-38.

Lee and Brown at JPL developed a leak monitoring system for the Cassini spacecraft that incorporated a model of the spacecraft<sup>8</sup>. However, constant leaks present a different FDI problem than the hard abrupt faults addressed here.

Since any motion-based FDI method is reliant on the accuracy of the measurement data used, great care has been taken here to get highly accurate angular and translational acceleration estimates from the gyros and accelerometers. These algorithms are presented in Ref. 3, which shares the nomenclature and spacecraft model used here.

## B. Range of applicability

This is a method for detecting and isolating fault modes in a system having a model describing its behavior and one or more measurements that are sampled regularly. In this method, said models are used to calculate past and present deviations from measurements that would result from said system with no faults, as well as from said system with one or more potential said fault modes. Algorithms that calculate and store these deviations, along with memory of when said faults, if present, would have an effect on the said actual measurements, are used to detect when a fault is present. Related algorithms are used to exonerate false fault modes and finally to isolate the true fault mode.

## C. Problem definition

Development of the FDI technology was driven initially through application in simulation to two specific spacecraft, the X-38 and the Mini AERCam. It has been refined and extended through hardware application for the MIT SPHERES experimental spacecraft, focusing primarily on maintaining high accuracy while streamlining and optimizing the algorithm for implementation in real-time software. Specifics regarding these applications are listed in this section.

As will be seen these spacecraft are all excellent candidates for motion-based thruster FDI. They are thruster-actuated, have relatively minimal sensing, and except for the X-38's thruster nozzle temperature sensors (which are actually not very useful in meeting the FDI requirements), do not have special-purpose thruster FDI sensors.

Motion-based thruster FDI such as this is dependent on an accurate dynamical spacecraft model, including accurate thruster and mass-property characterization. Mass-property uncertainty and variability is especially difficult for the X-38 or CRV due to the significant effects of fuel depletion (e.g., moves the CM forward by 20 cm as the De-orbit Propulsion Stage tanks empty) and unknown payload for the CRV, depending on the crew loading. This requirement motivated closely related research in on-line mass-property identification<sup>2,3</sup>.

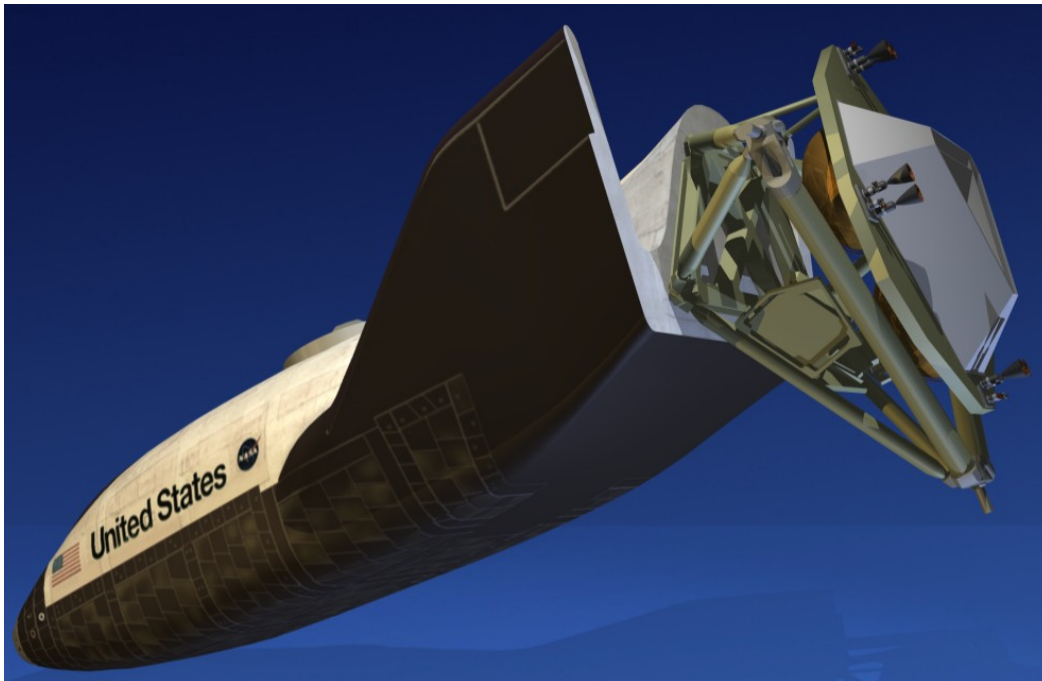


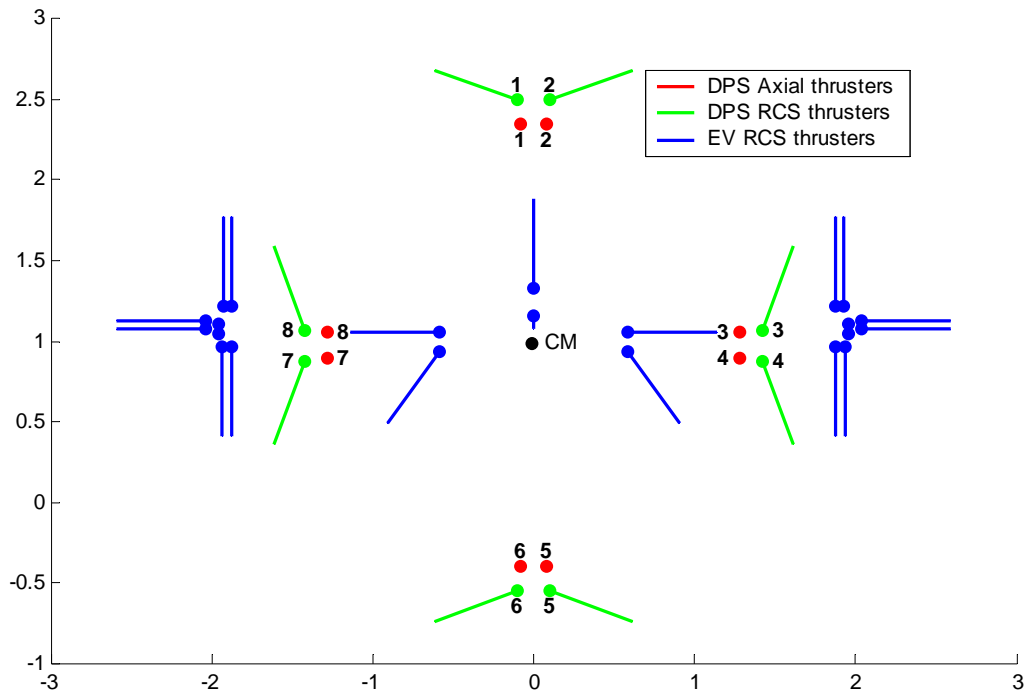
Figure 2. X-38, with entry vehicle and de-orbit propulsion stage

## 1. X-38

The Crew Return Vehicle (CRV) design consisted of a manned space vehicle, the Entry Vehicle (EV), based on a lifting body design, and a De-orbit Propulsion Stage (DPS). The CRV was designed to remain docked to the space station in a dormant mode until needed by the crew in an emergency. The X-38 (vehicle 201), shown in Figure 2, was to be the unmanned test vehicle for the CRV, until the cancellation of the X-38 program in 2002. Both vehicles were designed to maneuver on-orbit, de-orbit, and land using a large parafoil.

The DPS includes a set of axial and RCS thrusters fed by three mono-propellant hydrazine tanks. Although the CRV design used pressure sensors in the thrusters to detect faults, the X-38 had only temperature sensors, driving the need for motion-based thruster FDI.

The DPS has 8 axial thrusters (500 Newtons thrust level each) that fire along the longitudinal axis of the vehicle, providing the required de-orbit thrust for the 13,600 kg vehicle. During the 8-to-15-minute de-orbit burn, four, six, or eight of the axial thrusters fire continuously, controlled open loop, chosen symmetrically to produce minimal torque on the vehicle. The DPS also contains 8 reaction control system (RCS) thrusters (106 Newtons thrust level each) that are fired in sets of two or four by the attitude control system to control the roll, pitch, and yaw about the body axes. The EV has a completely separate set of RCS thrusters for use after DPS separation (the Attitude Control System or ACS thrusters); those are not considered here. Figure 3 is a rear-view schematic of the X-38 showing EV RCS, DPS RCS, and DPS axial thrusters. The axial thrusters fire directly back along the x-axis, and the DPS RCS thrusters fire in the y-z plane, with no x-axis component.



**Figure 3. X-38 thruster configuration, coordinates in vehicle structural frame [meters]**

Accelerometers and ring-laser gyros in the Honeywell Space Integrated GPS / INS (SIGI) are available for monitoring vehicle motions<sup>9</sup>. Temperature sensors in the thrusters provide fault information as well, but the response time limits the ability to detect faults sufficiently quickly so they were not used. They rise in about one second, but cool over a period of minutes. Thruster faults will be detected by comparing vehicle motions to the vehicle motions that would result if certain faults have occurred. Initially, only rotational accelerations were used, but the capability to use translational accelerations as measured by accelerometers has since been added to enable faster fault isolation and even more accurate FDI.

## 2. Mini AERCam

The Automation, Robotics, and Simulation Division of NASA Johnson Space Center is leading the development of the Miniature Autonomous Extravehicular Robotic Camera (Mini AERCam), a free-flying robotic inspection

vehicle also shown in Figure 1<sup>10,11,§</sup>. The Mini AERCam vehicle is designed for either remotely piloted operations or supervised autonomous operations including automatic station keeping, point-to-point maneuvering, and automated docking approaches. Being a highly autonomous free-flying spacecraft in close proximity to the Shuttle or ISS, thruster FDI (and mass-property ID to support it) is very important.

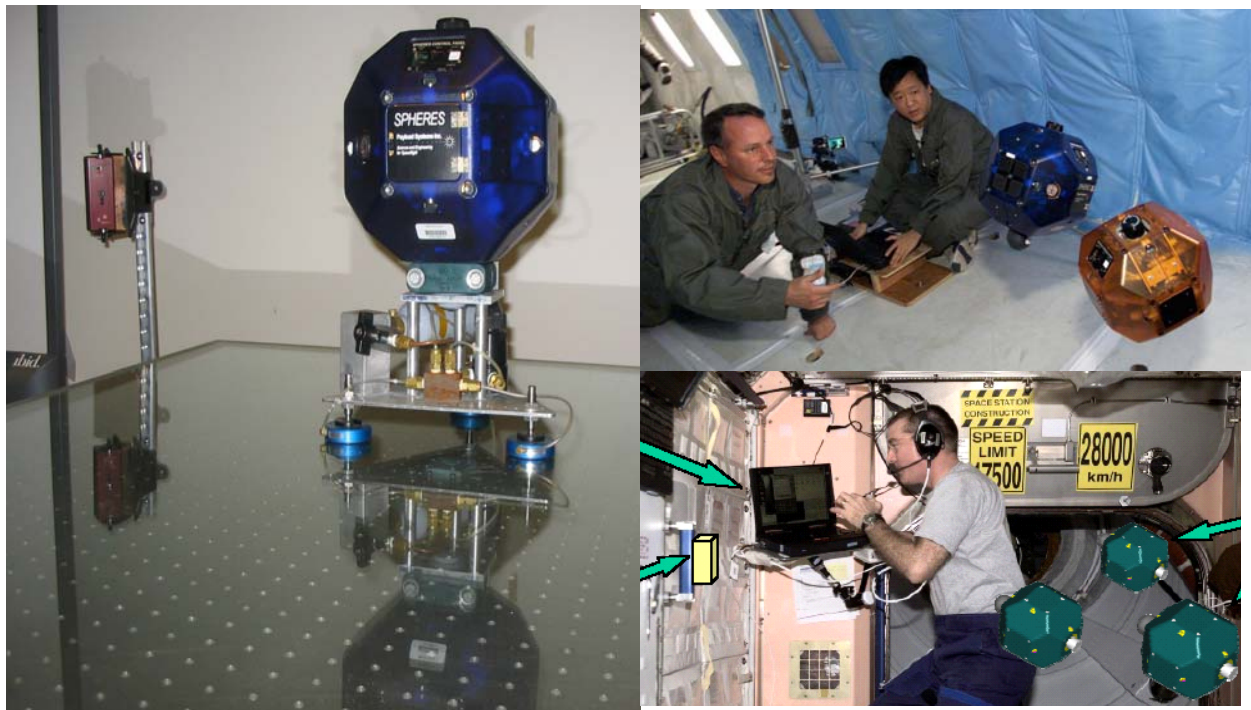
### 3. MIT SPHERES

SPHERES (Synchronized Position Hold, Engage, Reorient, Experimental Satellites) is a program developed by the MIT Space Systems Laboratory and Payload Systems, Inc. to provide MIT, government and other scientists with a long term, replenishable, upgradeable and cost-effective laboratory that exploits the microgravity conditions of space to approximate the dynamics presented by distributed satellite and docking missions<sup>12,13</sup>. The SPHERES testbed will be used for the validation of high risk control and autonomy technologies critical to the operation of these missions. SPHERES provides a low-risk, representative dynamic environment for the interactive development and verification of formation flight, rendezvous, and docking control and autonomy algorithms.

The SPHERES testbed consists of three micro-satellites which can control their relative positions and orientations in up to six degrees of freedom. The testbed can operate in 2-D (3-DOF motion) on a laboratory platform and in 3-D (6-DOF motion) on NASA's KC-135 and inside the International Space Station (ISS), as shown in Figure 4.

This experimental spacecraft is serving as the hardware platform for in-space validation of the ID technology. Its relevant characteristics are described in detail in Ref. 3 and summarized briefly here.

- 12 CO<sub>2</sub> thrusters controlling the 6 degrees of freedom
- Ultrasound-beacon-based global position and attitude determination
- Gyros and accelerometers on each axis
- RF communications
- Astronaut to conduct tests inside the ISS
- The flight software is written in C / Embedded C++, and runs on a Texas Instruments C6701 DSP.



**Figure 4. Experiments on NASA Ames air-bearing table; NASA KC-135; and as to be conducted on ISS**

Initial deployment in the ISS will include two satellite units and enough consumables for 20 hours of operations over a six month period. Astronauts will monitor the experiments on ISS and provide feedback to the scientists on

<sup>§</sup> <http://aercam.nasa.gov>

the ground approximately every two to four weeks. Using this feedback, scientists will enhance the performance of their controllers and send updated algorithms to ISS.

## II. Spacecraft model

All analysis has been done in inertial free space, so gravity is not included and earth-referenced frames such as LVLH have not been used. Gravity gradient torques, solar pressure, and aerodynamic drag are expected to be negligible for the spacecraft applications studied. However, if significant for other applications, they may be calculated and included by the ID algorithms as a modeled disturbance. In contrast to the treatment of these effects, the thrusters, being the principal actuators, will be modeled as accurately as possible, taking into account transient and multi-jet firing characteristics. The level of detail warranted is determined by the magnitude of the remaining uncertainty. In the spacecraft studied, major uncertainties included pulse-to-pulse thrust magnitude, vibration-induced sensor noise, and thruster misalignment.

Steps are taken throughout the derivations to enable a matrix representation, although the mathematical operations do not need to be done with matrix math. For example,  $F_{nom}$  and other terms are represented as diagonal matrices and matrix operations are used rather than elemental operations. The matrix-based construction of these and other equations enables a more mathematically compact representation, and facilitates the matrix manipulations that occur throughout the derivations. Alternatively and perhaps more directly, the expression for  $F_k$  could, for example, be written as for  $i = 1$  to  $N$ ,  $F_k(i) = S_k(F_{nom}(i) + F_{bias}(i) + F_{random,k}(i))T_k(i)$ , where  $F_{nom}$ , etc. are represented minimally and logically as vectors rather than diagonal matrices.

A more complete description of the many steps leading to the equations of motion and thruster models is presented in Ref. 3. Consistent nomenclature is used between these two research efforts.

### A. Equations of motion

Starting with Euler's dynamical equation, and assuming the spacecraft inertia tensor is constant, the rotational equations of motion (EOM) are\*\*

$$\dot{\omega} = I^{-1}(\tau - \omega \times (I\omega)). \quad (1)$$

The translational equations of motion are derived from Newton's second law of motion,

$$f = m\ddot{x} \text{ or } \ddot{x} = m^{-1}f, \quad (2)$$

The torque in Equation (1) and force in Equation (2) are set by the thruster firings as described below.

### B. Thruster force and torque models

$F_k$ , the thrust magnitude from each individual thruster at each control update, can be expressed as the following function, which accounts for:

- Thruster command – i.e., whether a thruster is commanded to fire at this control update
- Thruster faults – whether thrusters have failed off or on
- Thruster transient effects – latency when thrusters are turned on or off, as well as transient variations in thrust are modeled
- Reduction in thrust due to blowdown
- Reduction in thrust due to multiple thrusters firing
- The nominal thruster magnitude
- A constant random bias added to the nominal value
- A random pulse-to-pulse offset added to the nominal value

$$F_k = S_k(F_{nom} + F_{bias} + F_{random,k})T_k \quad (3)$$

---

\*\* Complete derivations are presented in Ref. 3.

where  $S_k$  accounts for blowdown in some systems and for the reduction in thrust when multiple thrusters fire (generally, regulated cold gas systems). For the X-38,  $B_k$  accounts for the reduction in thrust as the propellant tank(s) empties. For regulated cold gas systems,

$$S_k = g(\sum T_k). \quad (4)$$

In this case,  $S_k$  accounts for the reduction in thrust when multiple thrusters fire, and is represented as a function of the total number of thrusters firing at a given update time. This physical effect is primarily present in regulated cold-gas propulsion systems, where the pressure drop across the regulator increases as the number of thrusters increases.

The modeling and estimation approaches taken here analyze the acceleration that results over the control period when thrusters are fired. The computations are simplified by making the assumption that the thrusters produce a thrust that is constant during each sample period. In practice, latency, build up, trail off, and possibly thrust ringing are not negligible and can be modeled as in Ref. 14.  $T_k$  indicates the effective amount that each thruster is on (nominally ranges from 0.0 to 1.0), accounting for whether the thruster is commanded to fire, whether it has failed on or off, and the history of preceding thruster firings. It is calculated using the following function,

$$T_k = v(T_{command,1}, \dots, T_{command,k}, T_{failed\ off,k}, T_{failed\ on,k}). \quad (5)$$

The net force through the CM due to thrusters, rotated from the body to the inertial frame, can be calculated using the forces and directions of each thruster and the direction cosine matrix,  $R$ . A force disturbance term is also added (defined in inertial space), resulting in the total force on the vehicle. This force disturbance is defined as acting on the true CM, so there is no net moment produced. Effects like aerodynamic drag, gravity gradient, and solar pressure would be included in the disturbance models,  $\tau_{disturb}$  and  $f_{disturb}$ .

$$f = RDF_k + f_{disturb} \quad (6)$$

The moment due to thruster firings at time  $k$ ,  $M_{thrusters,k}$ , referenced to the vehicle's center of mass (the true CM, which is not known exactly) and defined in the vehicle frame is modeled as

$$M_{thrusters,k} = (L \times D)F_k. \quad (7)$$

Each column of  $L$  and  $D$  contains the information for an individual thruster, and the cross product shown here indicates that the  $i^{th}$  column of  $L$  is crossed with the  $i^{th}$  column of  $D$  and entered into the  $i^{th}$  column of the resulting [3-by- $N$ ] matrix.

Adding the disturbance terms and using  $R^{-1}$  to rotate from inertial to body frames, the resulting torque about the CM and force through the CM, measured in the body frame, can now be written.

$$\begin{bmatrix} \tau \\ f^{body} \end{bmatrix} = \begin{bmatrix} L \times D \\ D \end{bmatrix} F_k + \begin{bmatrix} \tau_{disturb} \\ R^{-1} f_{disturb} \end{bmatrix} \quad (8)$$

### C. Combined equations of motion

Combining the above results, rotational and translational motions are governed by Equations (9) and (10). These are written for use when the gyros and accelerometers provide measurements in the body frame (gyros give  $\omega$ , accelerometers give  $\ddot{x}^{body}$ ). These are the equations to be used in FDI development.

$$\dot{\omega} = I^{-1}((L \times D)S_k(F_{nom} + F_{bias} + F_{random,k})T_k + \tau_{disturb} - \omega \times (I\omega)) \quad (9)$$

$$\ddot{x}^{body} = m^{-1}(DS_k(F_{nom} + F_{bias} + F_{random,k})T_k + f_{disturb}^{body}) \quad (10)$$

In some cases, FDI accuracy will be sensitive to uncertainty in the model parameters. In those cases, mass or thruster-property identification may be used<sup>3</sup>.

The FDI algorithms are based fundamentally on analysis of the vehicle accelerations during thruster-firing periods. Consequently FDI accuracy is dependent on and limited by the accuracy of the velocity and acceleration estimates. Special purpose algorithms to do this accurately and efficiently are presented in Ref. 3.

The FDI assumes that the thrust is constant during each control sample period (although faults can occur abruptly), and the algorithms require estimates of the *average* accelerations over these control sample periods, to match with the *average* nominal thruster force over the corresponding periods, accounting for blowdown, transients, etc.

Any modeling of CMGs or RWAs would also appear in these equations if present on the spacecraft.

### III. Fault detection and isolation

It is generally true in system identification (ID) or FDI systems that reducing the degrees of freedom to be considered to the minimal level or otherwise constraining the problem will improve identification or detection performance. As discussed later, some alternative approaches were initially used on this problem that attempted to identify the strengths of the un-failed thrusters as well as finding the faults. This approach worked well on a simplified version of the X-38 application, but became unreliable when all 16 thrusters were simulated; both on and off faults were considered; mass properties were allowed to vary within tolerance; and in the presence of gyro noise. This led to the approach described below, which solves the problem taking full advantage of the problem statement—namely that only a single fault mode (which may include multiple-jet faults) from a pre-determined, finite list of candidates can be present, and that it will appear abruptly and persist.

#### A. Summary of the algorithm

At every control/FDI update, the disturbing acceleration,  $\hat{a}_{disturbing}$ , is calculated. This is the difference between the measured acceleration and that predicted by the equations of motion. This vector is compared with the vector of disturbing accelerations corresponding to each possible fault mode,  $a_{disturbing,i}$  (the disturbing acceleration one would expect if fault mode  $i$  were true). After a fault is detected, and once a clear match is found (the likelihood is sufficiently higher than all other possibilities), the fault mode is isolated. Specifics regarding filtering and other calculations follow.

#### B. Cataloging fault modes

An engineering analysis of the likely fault modes is made, for example, using a Failure Modes Effects Analysis (FMEA). Typically, depending on the criticality and likelihood of single-point-of-failure fault conditions (where that single point of failure may result in multiple thruster faults), this list is generated, balancing the desire to accommodate every possible fault condition (this is virtually unlimited) with the difficulty thereby imposed on the FDI system. The result is a list of “possible fault modes.” The FDI system will then consider that the only possible conditions are the case of no faults, or one of these “possible fault modes.”

It may be possible to use this analysis to assign an *a priori* likelihood of each fault mode, which would then be useful in FDI; however, for the applications studied here, there was no significant difference in *a priori* likelihood, and the algorithm was not developed to incorporate this information.

$a_{disturbing,i}$  is pre-calculated for every possible fault mode,  $i$ . Multiple-jet fault modes require further cataloging of each combination of thrusters that may be active (for example, if jets 1 and 2 fail off simultaneously, but are not always commanded to fire together, the disturbing acceleration will be different for the three different firing permutations: #1 firing, #2 firing, and both #1 and #2 firing).

This cataloging is done pre-flight, and the values may be updated periodically based on the updated state of the blowdown (the nominal strength of all thrusters drops as the tanks empty) or updated mass- and thruster-property estimates.

#### C. Estimating disturbing acceleration

This step calculates  $\hat{a}_{disturbing}$ , the disturbing acceleration vector attributed to the disturbance caused thruster faults. It is calculated as the difference between the acceleration estimate based on measurements, and the acceleration expected based on the physical spacecraft model and thruster commands.

$\alpha_{\text{known-system}}$  and  $\ddot{x}_{\text{known-system},k}^{\text{body}}$  are the physical-model based estimates of what the angular and translational accelerations should be. They are calculated assuming no faults are present and all physical parameters are at their known values (nominal or identified, but not necessarily true). Starting with the equations of motion (repeated here<sup>††</sup>), making appropriate substitutions, and assuming translational accelerations are measured in the body frame, the nominal accelerations are expressed as

$$\dot{\omega} = I^{-1}((L \times D)S_k(F_{\text{nom}} + F_{\text{bias}} + F_{\text{random},k})T_k + \tau_{\text{disturb}} - \omega \times (I\omega)) \quad (9)$$

$$\ddot{x}^{\text{body}} = m^{-1}(DS_k(F_{\text{nom}} + F_{\text{bias}} + F_{\text{random},k})T_k + f_{\text{disturb}}^{\text{body}}) \quad (10)$$

$$\alpha_{\text{known-system},k} = \hat{I}^{-1}((\hat{L} \times D)S_k(F_{\text{nom}} + \hat{F}_{\text{bias}})T_{\text{command},k} + \hat{\tau}_{\text{disturb}} - \omega \times (\hat{I}\omega)) \quad (11)$$

$$\ddot{x}_{\text{known-system},k}^{\text{body}} = m^{-1}DS_k(F_{\text{nom}} + \hat{F}_{\text{bias}})T_{\text{command},k} + \hat{f}_{\text{disturb}}^{\text{body}} \quad (12)$$

where the  $\hat{I}^{-1}$  in Equation (11) indicates the identified value of  $I^{-1}$  (rather than the inverse of the estimated value of  $I$ ). For the  $\omega \times (\hat{I}\omega)$  term (which will be insignificant for low angular rates, and may be negligible) either  $I_{\text{nom}}$ ,  $\hat{I}$ , or the inverse of the identified value of  $\hat{I}^{-1}$  may be used. Nominal or identified values for  $\hat{L}$ ,  $D$ ,  $S_k$ ,  $m$  are used, and measured values for  $\omega$  are used.  $\hat{I}^{-1}$ ,  $\hat{I}$ ,  $\hat{L}$ , and  $\hat{F}_{\text{bias}}$  are used in the equations (identified vs. nominal values) to facilitate integration with the accompanying parameter estimation research<sup>3</sup>, but the  $\hat{\cdot}$  could have been left off if on-line estimation is not considered. Estimates of the (known, unrelated to thruster faults) disturbing force and torque vectors may be used if available, as shown. The disturbing accelerations can then be calculated.

$$\hat{\alpha}_{\text{disturbing}} \triangleq \hat{\alpha} - \alpha_{\text{known-system}} \quad (13)$$

$$\hat{x}_{\text{disturbing}}^{\text{body}} \triangleq \hat{x}^{\text{body}} - \ddot{x}_{\text{known-system}}^{\text{body}} \quad (14)$$

Combining these two 3-vectors into a single 6-vector,

$$\hat{a}_{\text{disturbing}} \triangleq \hat{a} - a_{\text{known-system}} = \begin{bmatrix} \hat{\alpha}_{\text{disturbing}} \\ \hat{x}_{\text{disturbing}}^{\text{body}} \end{bmatrix} \triangleq \begin{bmatrix} \hat{\alpha} - \alpha_{\text{known-system}} \\ \hat{x}^{\text{body}} - \ddot{x}_{\text{known-system}}^{\text{body}} \end{bmatrix} \quad (15)$$

If both gyros and accelerometers are present (or some other method for estimating angular and translational accelerations), then FDI can proceed using all 6 degrees of freedom. As implemented, the X-38 simulation can use all 6 dof; the Mini AERCAM simulation uses only gyros, since those are sufficient; and SPHERES experiments use two options – (1) only gyros and (2) gyros and accelerometers.

#### D. Collecting, windowing, and filtering measurements for individual fault modes

If the signal-to-noise ratio were high enough and each fault mode signature ( $a_{\text{disturbing},i}$ ) were unique (or sufficiently separable in the noise present), maximum likelihood FDI analysis of the  $\hat{a}_{\text{disturbing}}$  readings could be carried out on the values at each time step, as was done in Ref. 5.

However, in the X-38 application studied here, sensor noise, mass property variations, and fault modes with similar  $a_{\text{disturbing},i}$ , require several extensions from this fundamental work:

- **Collecting** relevant similar measurements prior to filtering – for example, collecting  $\hat{a}_{\text{disturbing}}$  estimates for cases when only jet #3 was commanded to fire.
- **Windowing** these collections of measurements to enable detection of an abruptly appearing thruster fault, while also allowing accurate isolation once a fault is detected.

<sup>††</sup> In this paper, repeated equations are listed with the original equation number, in italics.

- **Filtering** the collected measurements is performed to reduce the effect of noise. Averaging (mean) has been found to work fine if the window is selected properly.
  - **Fault isolation post-processing** of the maximum likelihood parameters
- Collecting, windowing, and filtering are discussed in this section.

### 1. Collecting

The need for collection is highlighted by the case in which a failed-off thruster might be commanded to fire infrequently for short durations. The information about the fault may be detectable if this data is collected and examined together, whereas it may not be sufficiently persistent to be detected by a more conventional approach. Collection is implemented as follows: the single  $\hat{a}_{disturbing}$  vector is stored at all times; the history of each fault mode being *active* or *inactive* (and sub-mode information for the multi-jet case) is also stored.

### 2. Windowing

The problem statement says that thruster faults will appear abruptly. To maximize the noise reduction from filtering, it would be good to maximize the window size. However, this will compromise the detection of abrupt faults – if the filter uses data from before the fault occurred, as well as afterwards. Optimal selection of the window size used for fault detection requires consideration of the noise reduction needed, the acceptable time before detection, as well as the relative costs of false positives and failed detection (risk optimization).

For the X-38, a window size of 10 (equal to one second) was found to provide a good balance between speed of response and accuracy. Also, a minimum number samples (5 for the X-38) is required before maximum likelihood FDI analysis is allowed to proceed for a given fault mode.

The windowing situation changes slightly following detection. Since the problem statement says that only a single fault mode (possibly affecting multiple jets) will appear, once it is detected, the window can begin at the time of detection and grow without bound. To gain a few extra samples, conservative analysis can allow a number of samples before detection, reasoning that the fault was present for some time prior to detection. These extra samples may be important in allowing fast fault isolation.

However, the calculation of  $\lambda_{inactive,i}$  following detection keeps a finite sliding window. The rationale for this may become clearer when  $\lambda_{inactive,i}$  is discussed, but basically this measurement is intended to spike up when the true fault becomes active, thus allowing this fault to be exonerated. Since the true fault mode is not known, the best window for  $\lambda_{inactive,i}$  calculation cannot be known. The best that can be done is to continue to use a sliding window and to hope that the true fault will appear for long enough during that window to raise  $\lambda_{inactive,i}$  above the prescribed threshold and allow for fault  $i$  to be exonerated.

### 3. Filtering

Once the data has been collected and windowed, it is filtered to reduce the effects of noise in the calculation of the likelihood parameters. An average (mean) is taken, and found to work well. This is a modification of the conventional maximum likelihood approach, as discussed in Section III.E.2. Since it is known that faults will occur abruptly, this windowing and filtering method is preferred over an IIR (e.g., exponential) filter that would carry through information for longer. The implementation as described here avoids introducing any phase lag between the cause (thruster firings) and effect (vehicle motions), as would be introduced by a linear infinite impulse response filter or Kalman filter that would bias the FDI.

As mentioned earlier, one of the challenges of FDI for systems with on-off actuators is that faults are only observable when *active*. For example, “off” faults are observable only when the jets are commanded to fire. For each fault mode, only the relevant  $\hat{a}_{disturbing}$  measurements are used to calculate  $\hat{a}_{disturbing,active,i}$  and  $\hat{a}_{disturbing,inactive,i}$ , which are in-turn used to calculate  $\lambda_{active,i}$ ,  $\lambda_{inactive,i}$ , and  $\lambda_{active,i0}$ . So  $\hat{a}_{disturbing,active,i}$  is calculated as the mean over a window of  $\hat{a}_{disturbing}$  data during which fault mode  $i$  was active (for the X-38 and Mini AERCam, when jet  $i$  is commanded to fire), and  $\hat{a}_{disturbing,inactive,i}$  uses  $\hat{a}_{disturbing}$  data during which fault mode  $i$  was not active (when jet  $i$  is not commanded to fire).

## E. Maximum likelihood

Although the acceleration estimator is nonlinear and sub-optimal<sup>‡‡</sup>, it is reasonable to assume<sup>§§</sup> that the estimated disturbing acceleration readings,  $\hat{a}_{disturbing}$ , are normally distributed about the disturbing acceleration values,  $a_{disturbing,*}$  corresponding to the true fault mode. So the probability density for the true disturbing acceleration values,  $a_{disturbing,*}$ , conditioned on the measurement history  $M$ , is<sup>5,15</sup>

$$p(a_{disturbing,*} | M) = (2\pi)^{-n/2} |P_a|^{-1/2} e^{-\frac{1}{2}([a_{disturbing,*} - \hat{a}_{disturbing}]^T P_a^{-1} [a_{disturbing,*} - \hat{a}_{disturbing}])} \quad (16)$$

where  $n$  is the number of degrees of freedom used in the FDI, and  $P_a$  is the estimation error covariance of the disturbing acceleration.

### 1. Estimation error covariance of the disturbing acceleration

The  $P_a$  matrix determines how close the measurements are expected to match the predicted disturbing accelerations. How close a match can be expected is based on the sensor noise (and subsequent accuracy of  $\hat{a}_{disturbing}$ ) as well as uncertainties in the nominal or failed system models used to generate  $a_{disturbing,i}$ .

In some cases, setting it accurately is almost irrelevant. If all sensors have comparable accuracy (e.g., x, y, and z-axis gyros are the same model), and the uncertainty in system models is small,  $P_a$  may be set as an identity matrix. With the sensors the same, and uncorrelated, and with negligible contribution from model uncertainty, a diagonal matrix with equal diagonal elements will work. As will be discussed later, the detection is based on a ratio of likelihoods, and the isolation thresholds will need to be set anyway, so having  $P_a$  as an identity matrix will work.

If different sensors are used, (for example, gyros and accelerometers) it may be sufficient to have the matrix be diagonal, but with different values for the angular acceleration estimates vs. the translational acceleration estimates.

However, if model uncertainty is significant, as compared with estimation error, it may warrant more careful calculation of  $P_a$ . In this case,

$$P_a = P_{\hat{a}_{disturbing}} = P_{\hat{a}} + P_{a_{known-system}} \quad (17)$$

where  $P_{\hat{a}}$  primarily represents the uncertainty due to the sensors and  $P_{a_{known-system}}$  represents the contribution of model uncertainty. This shows the variance summation due to Equation (15).

As an example, consider the case of very accurate estimation, but inaccurate modeling of thruster and mass properties, and in particular the overall thruster force scale factor (due to blowdown or multi-jet firing). This extension of  $P_a$  calculation was driven by effects noticed during the SPHERES implementation that had the root cause in the inaccuracy in the multi-jet thrust reduction model.

In this case,  $P_a$  would equal  $P_{\hat{a}}$ , and be very small when no thrusters are firing (either because none were commanded to fire, or if one was commanded to fire and it had failed off). This analysis assumes no systematic estimation errors, as would be caused by mis-calibration or other effects that could result in a non-Gaussian estimation error distribution. If systematic errors existed, a more detailed approximation of  $P_{\hat{a}}$  would be required.

When thrusters fire, the uncertainty in mass and thruster properties (e.g., mass, mass center, inertia, thruster strength [both constant and pulse-to-pulse], direction, and location) will result in a  $P_{a_{known-system}}$  that is different

---

<sup>‡‡</sup> The maximum-likelihood analysis to follow relies on an assumption that the noise is zero-mean, un-correlated, and Gaussian. For a linear system and an optimal estimator, this could be true, but for the system here, it is not. The most significant deviation from this condition results from inaccurate mass-property information in the system model, which causes biased estimation of the disturbing acceleration. The approach taken here is to still use the maximum-likelihood approach, but to address problems with the basic assumptions – by averaging data and identifying mass properties to reduce estimation bias.

<sup>§§</sup> This simplifying assumption is common in maximum likelihood analysis, and was made also by [Deyst 1976].

depending on the thrusters assumed to fire (a function of the thruster command pattern and the fault mode). Due to the inverse operation, and multiplications involved, the true distribution will not be Gaussian, but it is reasonable to ignore this fact and attempt to approximate the equivalent  $P_{a_{\text{known-system}}}$ . This could be done for each possible firing pattern in simulation, or by the following approximate method:

$$P_{a_{\text{known-system}},i,k} \approx N_k P_{a_{\text{known-system}},\text{per-thruster}} + P_{a_{\text{known-system}},\text{scale-factor},i,k} \quad (18)$$

where  $N_k$  is the number of thrusters firing at time step  $k$ , accounting for both the thruster command and the fault mode under consideration,  $i$ . This assumes that the variance is approximately proportional to the number of thrusters firing, which is likely to be true if the thrusters are symmetric (e.g., produce equal acceleration magnitudes, as for the MIT SPHERES or Mini AERCam). The  $P_{a_{\text{known-system}},\text{scale-factor},i,k}$  term accounts for a particular component of uncertainty – an error in the model of thruster scale factor. Since it scales all thrusters equally, it will scale the acceleration vector as well. This will create an uncertainty in a single direction, defined by the vector from the origin to the nominal location in acceleration space (rotational or translational).

## 2. Combining measurements at multiple times

Following maximum likelihood theory, when multiple, independent measurement samples are considered, the probabilities *multiply*:

$$\begin{aligned} p(a_{\text{disturbing},i} | M) &= \prod_{k_{\text{window}}} (2\pi)^{-n/2} |P_a|^{-1/2} e^{-\frac{1}{2}(a_{\text{disturbing},i} - \hat{a}_{\text{disturbing}})^T P_a^{-1} (a_{\text{disturbing},i} - \hat{a}_{\text{disturbing}})} \\ &= (2\pi)^{-nk_{\text{max}}/2} |P_a|^{-k_{\text{max}}/2} \prod_{k_{\text{window}}} e^{-\frac{1}{2}(a_{\text{disturbing},i} - \hat{a}_{\text{disturbing}})^T P_a^{-1} (a_{\text{disturbing},i} - \hat{a}_{\text{disturbing}})} \\ &= (2\pi)^{-nk_{\text{max}}/2} |P_a|^{-k_{\text{max}}/2} e^{-\frac{1}{2} \sum_{k_{\text{window}}} (a_{\text{disturbing},i} - \hat{a}_{\text{disturbing}})^T P_a^{-1} (a_{\text{disturbing},i} - \hat{a}_{\text{disturbing}})} \\ &= (2\pi)^{-nk_{\text{max}}/2} |P_a|^{-k_{\text{max}}/2} e^{-\frac{1}{2} \lambda_{\text{active},i}} \end{aligned} \quad (19)$$

where the summation over  $k_{\text{window}}$  indicates the use of measurements at different times, with the size of this window determined as described above.

Given disturbing acceleration measurements,  $\hat{a}_{\text{disturbing}}$ , and knowing the disturbing acceleration values corresponding to each possible fault mode,  $a_{\text{disturbing},i}$ , the most likely fault mode is found by finding the  $a_{\text{disturbing},i}$  that maximizes this probability density function. The subscript  $i$  indicates the fault mode number corresponding to the disturbing acceleration. This function is maximized when the likelihood argument,  $\lambda_{\text{active},i}$ , in the following expression is minimized:

$$\lambda_{\text{active},i} = \sum_{k_{\text{window}}} \left( a_{\text{disturbing},i} - \hat{a}_{\text{disturbing,active},i} \right)^T P_a^{-1} \left( a_{\text{disturbing},i} - \hat{a}_{\text{disturbing,active},i} \right) \quad (20)$$

$\hat{a}_{\text{disturbing,active},i}$  is the estimated disturbing acceleration for a sample during which fault mode  $i$  was *active*. For example, fault mode #1 on the X-38 corresponds to RCS jet 1 being failed off, and  $a_{\text{disturbing},1}$  is  $[-0.0142 \text{ rad/sec}^2, -0.0013 \text{ rad/sec}^2, 0.0040 \text{ rad/sec}^2, 0 \text{ m/sec}^2, -0.0070 \text{ m/sec}^2, -0.0025 \text{ m/sec}^2]$ . This means that if RCS jet 1 is commanded to fire, and it has failed off,  $\hat{a}_{\text{disturbing}}$  should equal  $a_{\text{disturbing},1}$ .

However, this approach was not found to work well in simulation on the X-38 application, and the slightly modified alternative, described previously was found to work very well. In the approach used, the  $\hat{a}_{\text{disturbing}}$  measurements from each sample in the window are *first* averaged together, and *then* Equation (16) is applied, which

is equivalent to Equation (19) calculated for a single measurement. The subtle difference is the difference (unconcerned with scale factors here) between Equation (20) and

$$\lambda_{active,i} = [a_{disturbing,i} - \frac{1}{N_{window}} \sum_{k_{window}} \hat{a}_{disturbing,active,i}]^T P_a^{-1} [a_{disturbing,i} - \frac{1}{N_{window}} \sum_{k_{window}} \hat{a}_{disturbing,active,i}]. \quad (21)$$

It appears that Equation (21) performs better than the theoretically correct one because it better accounts for the imperfection in the problem statement. The theoretically correct option from Equations (19) and (20) considers that each separate measurement is independent of one another and unbiased. However, in reality, in addition to the zero mean noise sources, several effects such as mass- and thruster-property uncertainty cause biased measurements. Given this bias, it seems logical to average the biased measurements into a single measurement vector before comparing them to  $a_{disturbing,i}$ , and it works well, so this is what is done.

This modification relies on  $P_a^{-1}$  being the same for all values in the measurement window. In the applications studied, this limitation was not a problem, since for the X-38,  $P_a^{-1}$  was constant, and for SPHERES, although  $P_a^{-1}$  is not constant, windowing is not needed.

If  $P_a^{-1}$  is diagonal (as it commonly is assumed to be), the calculation may be simplified by bringing the  $P_a^{-1}$  term out of the summation. Since different  $\lambda$ 's are calculated using different numbers of samples, each  $\lambda$  is normalized by dividing by the number of samples used to calculate it.

This expression is calculated and used both to detect and to isolate faults. When fault mode  $i$  is true, there will be a close match between  $a_{disturbing,i}$  and  $\hat{a}_{disturbing,active,i}$ , resulting in a low value for  $\lambda_{active,i}$ . Calculation of  $\lambda_{active,i}$  may be halted once a fault mode has been isolated. Alternatively, it may continue, ensuring validation of the isolation decision, or the FDI process may be re-initialized following isolation to detect and isolate further faults.

The approach taken in Ref. 5 basically makes this above calculation of  $\lambda_{active,i}$ , without the preceding collection, windowing, and filtering. It is then used directly for fault detection and isolation, without the following  $\lambda_{inactive,i}$  calculations and additional post-processing.

The likelihood argument,  $\lambda_{inactive,i}$ , is calculated using data from periods where the fault mode was not *active*, as follows:

$$\lambda_{inactive,i} = \sum_{k_{window}} (a_{disturbing,inactive,i} - \hat{a}_{disturbing,inactive,i})^T P_a^{-1} (a_{disturbing,inactive,i} - \hat{a}_{disturbing,inactive,i})^{***}. \quad (22)$$

where  $a_{disturbing,inactive,i}$  is the disturbing acceleration that would be expected if fault mode  $i$  were true, when fault mode  $i$  is inactive. This is zero, allowing for simplification to

$$\lambda_{inactive,i} = \sum_{k_{window}} (\hat{a}_{disturbing,inactive,i})^T P_a^{-1} (\hat{a}_{disturbing,inactive,i}). \quad (23)$$

$\lambda_{inactive,i}$  is used to declare fault modes false following detection and prior to isolation, so it does not need to be calculated at other times.

When fault mode  $i$  is true,  $\lambda_{inactive,i}$  will have a low value, just as  $\lambda_{active,i}$  does. However, when fault mode  $i$  is not true, it will have a high value whenever the true fault mode is *active*, and a low value at other times. The  $\lambda_{inactive,i}$  values are monitored for spikes, and when the value exceeds a threshold, fault mode  $i$  is decided to be false from that point forward (this processing occurs only after fault detection).

---

\*\*\* This and the remaining likelihood-parameter calculations are written in the theoretically correct form, where the summation is taken over the individual measurement likelihood exponents, as in Equation (20). However, the preferred option is to first average measurements over the window, as discussed and presented in Equation (21).

The likelihood argument,  $\lambda_{active,i0}$ , is calculated using data from periods where fault mode  $i$  was *active*, as follows:

$$\lambda_{active,i0} = \sum_{k_{window}} \left( \mathbf{0} - \hat{\mathbf{a}}_{disturbing,active,i} \right)^T \mathbf{P}_a^{-1} \left( \mathbf{0} - \hat{\mathbf{a}}_{disturbing,active,i} \right). \quad (24)$$

$\lambda_{active,i0}$  measures the likelihood that  $\hat{\mathbf{a}}_{disturbing,active,i}$  could have resulted even though no faults (none at all, not just that fault mode  $i$  is not true) were present (that is why  $\alpha_{disturbing,i} = 0$ ). When fault mode  $i$  is true,  $\lambda_{active,i0}$  will have a high value, and when no faults are present, it will have a low value. The ratio between  $\lambda_{active,i}$  and  $\lambda_{active,i0}$  is monitored to detect faults, as described below.

## F. Fault detection

At each FDI update, for each possible fault mode,  $\lambda_{active,i}$  is evaluated using the windowed readings. The likelihood argument corresponding to no fault,  $\lambda_{active,i0}$ , is evaluated using the same windowed relevant readings, but with zero substituted for  $\alpha_{disturbing,i}$ . So  $\lambda_{active,i0}$  should be low (a close match) when no fault mode (including  $i$ ) is true. The ratio of likelihood arguments is calculated as

$$\lambda_{ratio,i} \triangleq \frac{\lambda_{active,i}}{\lambda_{active,i0}}. \quad (25)$$

A fault is detected when  $\lambda_{ratio,i}$  falls below a threshold; this is a generalized likelihood ratio test<sup>16</sup>. The calculation of  $\lambda_{active,i0}$  and use of the ratio in fault detection (as compared to using a fixed threshold for  $\lambda_{active,i}$ ) is less susceptible to an incorrect  $\mathbf{P}_a$  value and distortion by other fault modes (e.g., when fault mode  $j$  is active, it will affect calculations of  $\lambda_{active,i}$  and  $\lambda_{inactive,i}$ ). Further tests are then performed before isolating a particular fault mode, as described below. Evaluation of individual  $\lambda_{active,i0}$ 's for each fault mode is critically important – evaluating  $\lambda_{active,0}$  based on all (windowed) data may not indicate a fault if a failed-off thruster has not fired recently.

## G. Fault isolation

Immediately following fault detection, the finite list of possible fault modes is analyzed to see if any can be exonerated. For example, if a fault mode has never been active since the initialization of the FDI system, it can be exonerated at this time. Once a fault mode is exonerated, no further calculations of likelihood parameters are required.

Then, at each FDI update, the likelihood arguments,  $\lambda_{active,i}$  and  $\lambda_{inactive,i}$ , are calculated using all relevant data since the time of detection and compared to certain thresholds and to each other. If a fault mode is true, both  $\lambda_{active,i}$  and  $\lambda_{inactive,i}$  should be low, indicating the fault mode  $i$  fits the data well when it is both *active* and *inactive*. This is used first to remove fault modes from consideration – if  $\lambda_{active,i}$  or  $\lambda_{inactive,i}$  ever rise above a threshold (as they would whenever the data does not match well with fault mode  $i$  being true), fault mode  $i$  is decided to be false and removed from further consideration.

This additional step that allows for fault modes to be removed from consideration is important for improving the speed and accuracy of the FDI. In some cases, a fault is correctly identified by process of elimination – all other fault modes are exonerated. Analysis and removal of false fault modes that are similar to the true one enables more discriminating decision thresholds to be set, improving accuracy.

For a fault to be isolated,  $\lambda_{active,i}$  must be below a “low” threshold, indicating a very close match, while no other faults are below a higher threshold.

Some faults are virtually indistinguishable from one another in terms of the resulting  $\hat{a}_{disturbing}$ . For example, if only angular accelerations are used on the X-38, this set of four (referring to Figure 3): axial 1 off, axial 2 off, axial 5 on, axial 6 on<sup>†††</sup>. Even if the fault signatures,  $a_{disturbing,i}$ , are identical, it is still possible to distinguish one from another as long as the fault modes are not *active/inactive* in synch with one another.

Unfortunately, it is not uncommon for mode activations to be synchronized due to the control system. For example, consider two jets, A and B, that are physically opposite one another (i.e., the acceleration vectors are opposite). If A fails off it may be commanded to fire continuously since no control thrust is produced and B is not likely to be fired at all since it would be pushing in the opposite direction. In this situation, the fault modes for both jet-A-off and jet-B-on are active. So in the extreme limit where the nominal accelerations from A and B are identical and the activations are exactly synchronized, it is mathematically impossible (the information is not there) to distinguish between these two fault modes.

Another alias problem occurs in cases like on the X-38 axial thrusters in the example listed above. This problem occurs because those thrusters are fired open-loop in sets of 4, 6, or 8 at a time and sets of them have near-identical fault signatures.

For the X-38, the approach taken is to alter the axial firing pattern (e.g., changing from 1-2-3-5-6-7 on to 2-3-4-6-7-8 on) while maintaining symmetry. Since the firing pattern is adjusted to isolate the failed thruster, once the fault has been identified, the pattern is left in a state that makes the fault *inactive*, providing reconfiguration as well as FDI in this case. FDI-driven excitation of fault modes such as this example is generally valuable in expediting the isolation.

## H. Continued monitoring

Following isolation, one approach may be to assume that this was the only fault mode that will appear (as presented in the problem statements for the various spacecraft), and that the isolation was done correctly. This would enable the halting of all FDI monitoring calculations.

However, if further verification is desired, it is possible to continue to calculate and monitor the likelihood parameters (perhaps with modified windowing) to see if the isolation decision was incorrect.

Additionally, once a fault mode is isolated, it may be accepted as a true fact. At this point, the entire FDI monitoring system may be re-initialized with this knowledge, enabling detection and isolation of other fault modes.

## I. FDI based on RLS analysis

In an initial attempt at solving the FDI problem for the X-38, the authors used recursive least squares (RLS) analysis. As had been done in Ref. 7, thruster parameters were identified using an exponentially weighted RLS algorithm. While sufficient for thruster bias identification (with full knowledge of thruster hard faults), this approach did not provide sufficiently reliable FDI for the X-38 application for three main reasons:

- Relatively high noise levels were present (primarily due to gyro noise and pulse-to-pulse thruster variation).
- Exponential weighting meant that thrusters fired relatively sparsely (e.g., RCS thrusters as compared to axial thrusters) were not identified well
- Since multiple axial thrusters are fired continuously, observability of those parameters was very low.

A second, “targeted” RLS-based approach used multiple RLS algorithms, each one identifying the strength of a single thruster with the assumption that all other thrusters were operating nominally. This effectively addressed problems 2 and 3 above, but problem 1 remained. Also, the assumption that all other thrusters are nominal causes partial false positives when the failed thruster fires at the same time a good thruster fires. Methods were developed to address these remaining problems, but results were not sufficiently reliable, motivating development of the maximum-likelihood-based solution. However, the RLS approach should not be discounted, as it may work well on less challenging applications.

If such a method could be made to work for a given application, it has the clear benefit of being able to detect and isolate arbitrary fault configurations, removing reliance on the initial cataloging of all possible faults. It can also deal with soft faults, in which a thruster may output at a reduced thrust level.

---

<sup>†††</sup> The on vs. off fault modes in this example could be distinguished if translational accelerations were used for FDI, reducing the alias set from 4 to 2.

## J. FDI using a bank of Kalman filters

A well-studied approach in FDI is to use a bank of Kalman filters, one for each fault mode, plus one for the case of no faults. So for example, the KF for fault mode #1 would use a modified system model in which thruster #1 has failed off. This approach is commonly applied to detect sensor faults as well as actuator faults. In most systems in which actuators and sensors are continuously valued (as opposed to the on-off actuators present here), the KF prediction error (the difference between the state estimate before the measurement update and the measurement itself) is monitored for each of these KFs. These “residuals” or “innovations” should be zero-mean white noise if the model used in the KF accurately represents the actual system. So only one member of this bank of KFs should have residuals near zero – detecting which one does gives the result of the FDI.

However, this is complicated by the use of on-off actuators in thruster FDI. If a thruster has failed off, but is not commanded to fire, the KF for that fault mode will (correctly) have a low residual – but so will the case of no faults present and any other KFs whose faults are not presently *active*. Making use of this information is possible, but if intermittently *active* faults are to be addressed (such as a failed-off jet that is commanded to fire intermittently), it will require the windowing and collecting data step described in Section III.D, as well as the post-processing logic described in Sections III.F and III.G.

The initial intent when this approach was explored was that FDI would be able to make better use of the noisy gyro signals by using Kalman filtering. Unfortunately, one does not know which KF is to be believed until the FDI is done correctly, and the problems caused by intermittently *active* faults are present here as with the maximum-likelihood approach. For these reasons and the fact that the pre-filtering and post-processing logic had already been tuned for the maximum-likelihood FDI, the bank of Kalman filters approach was not pursued beyond the initial implementation and testing. It could provide a valuable alternative in certain thruster FDI applications, depending on the sensors, thruster transient characteristics, etc. Results comparing this approach with the maximum-likelihood approach are presented in Section IV.B.

## K. Efficiency, extensions

Many of the terms needed in this analysis, such as  $a_{disturbing,i}$ , can be pre-computed (as done for the SPHERES hardware implementation) or updated periodically. The algorithm is then relatively efficient. It scales better than linearly as more fault modes are added, since some information is shared between analyses of different fault modes (e.g., estimating  $\hat{a}_{disturbing}$ ).

The FDI is independent of the acceleration estimation method or the sensors used (e.g., gyros vs. sun sensor or video image analysis), although it is important that the time-correlation between periods of thruster firings and the resulting accelerations is not distorted.

Any combination of rotational or translational accelerations may be used—e.g., all gyro-based, all gyro- and accelerometer-based. If more sensors are used, better discrimination between faults is possible since the comparison space is of higher dimension. In this case, it is important that  $P_a$  is set appropriately to balance the relative influence of each acceleration estimate.

If reaction wheel assemblies (RWAs) are present, their effect could be modeled and accounted for, allowing for thruster FDI to proceed. In general terms, a maximum-likelihood approach such as used here should also be able to detect and isolate RWA faults. However, the continuous nature of RWAs means that windowing is not necessary, and the FDI should be straightforward.

This algorithm has been applied successfully to several vehicles, demonstrating its generic applicability for a broad class of thruster controlled spacecraft.

## IV. Application to the X-38 and Mini AERCam

The ID and FDI algorithms were applied in MATLAB<sup>17</sup> simulation to the X-38 and Mini AERCam, and are described in this Section. In each section, only the aspects of the solution that are new for that particular test are discussed.

### A. X-38 thruster FDI

The maximum-likelihood thruster FDI algorithm was applied to the X-38, with 40 different fault modes simulated, including each of the 8 RCS and 8 axial thrusters being failed-off or failed-on (32 single-jet faults) and 4 pairs of RCS jets being failed off or on (8 multiple-jet fault modes). Every mode has been tested multiple times and

accurate detection and isolation generally occurs within 5 seconds. Fault detection usually takes only 0.5 seconds, and most faults are identified within about 2.0 seconds<sup>†††</sup>. The switching of axial thrusters with near-empty tanks to distinguish between similar fault modes in some cases causes the time for isolation to exceed 5 seconds. Tuning of the decision thresholds and logic can be done to trade off accuracy vs. speed. Presently this has been biased towards accuracy. If the signal-to-noise ratio varies, these decision thresholds could be varied as well to maintain decision optimality. In extended testing, a success rate of 99.9994% was measured. In those tests, a single test case included running a simulation, initiating a fault, and observing either a correct FDI or a failure.

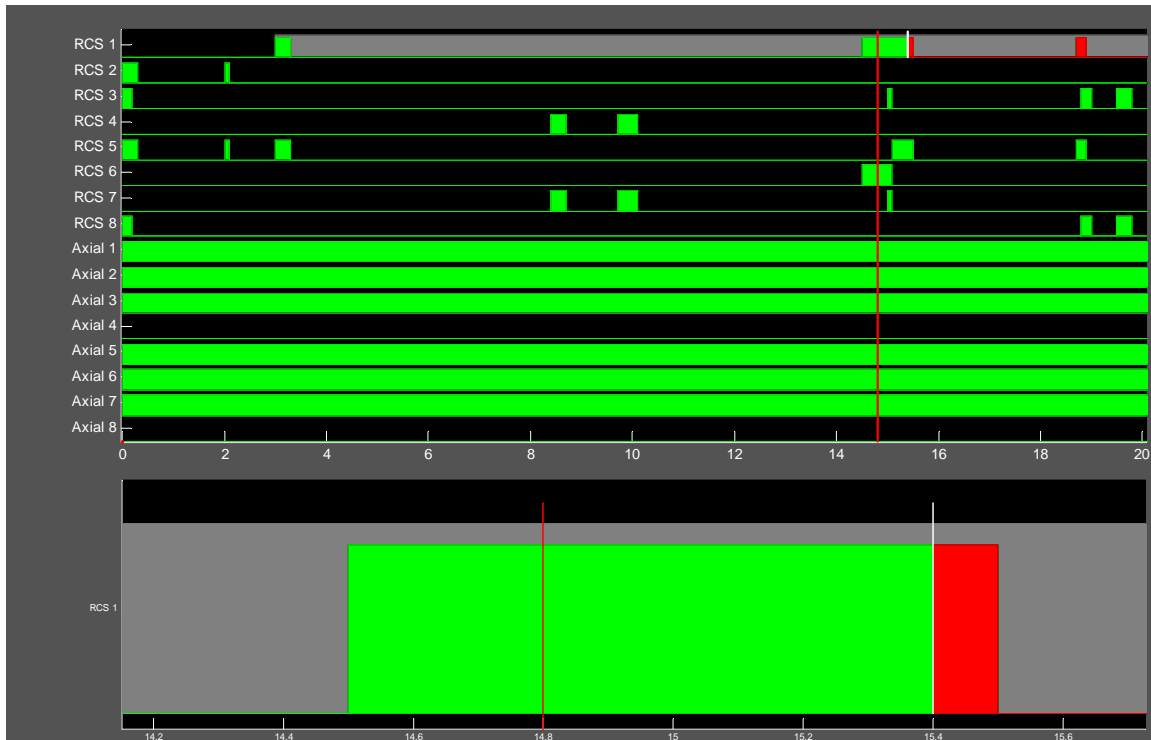
### 1. RCS jet 1 failed off

An example case is discussed here and shown in Figure 5 through Figure 10, for RCS jet 1 failed off. The first plot shows the thruster firing history during this 20-second run, in which the vehicle regulated automatically to an initial error and in response to disturbing torques from the axial thrusters<sup>§§§</sup>.

The first 8 rows in Figure 5 show the RCS jets pulsing to regulate attitude. The next 8 rows show that axial jets 1-2-3-5-6-7 were on continuously during this run. Below that is a zoomed in view of the detection and isolation of RCS jet 1 fault.

Figure 7 shows a legend corresponding to the thruster history as well as the animation figure.

Figure 8 shows a rear view of the vehicle with thrusters firing, torque monitors indicating the net torque produced by the axial and RCS thrusters, and the fault isolation result along with a visualization of the likelihood argument,  $\lambda_{active,i}$ .



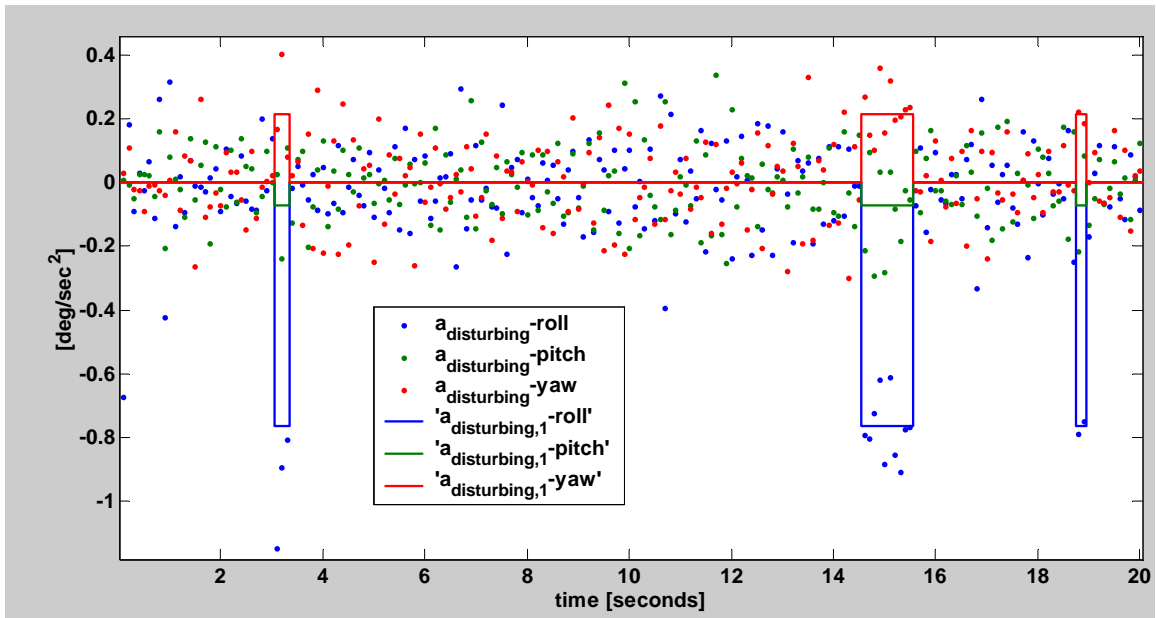
**Figure 5. Thruster command and fault history, FDI results**

The fault appeared abruptly at 3.0 seconds (it is a coincidence that jet 1 tried to fire at that time as well). Detection of the fault occurs at 14.8 seconds, indicated by the red vertical line in the plots. This detection came after a total of 0.6 seconds of firing (from 3.0-3.3 seconds and 14.5-14.8 seconds). The conservative isolation logic then

<sup>†††</sup> Since fault modes may not be observable depending upon whether their thrusters are commanded to fire, the detection and isolation times listed indicate the total duration for which the fault was active (fired for failed-off, or not fired for failed-on).

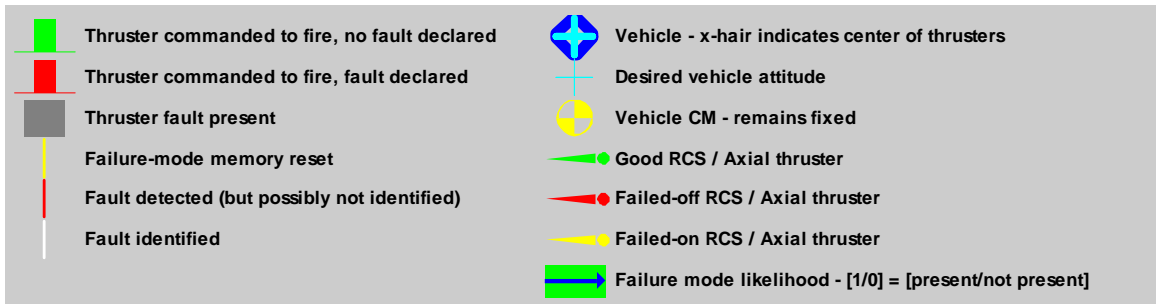
<sup>§§§</sup> The thrusters are fired in a nominally symmetric pattern, but since the center of the thrusters is not aligned exactly with the CM, and due to thruster biases and pulse-to-pulse variations, the axial thrusters create a significant torque disturbance.

waited another 0.6 seconds, until isolating the fault mode at 15.4 seconds, a total of 1.2 seconds of firing time. The isolation time is indicated by the vertical white line, and the fact that subsequent thruster commands are indicated by red rectangles rather than green.



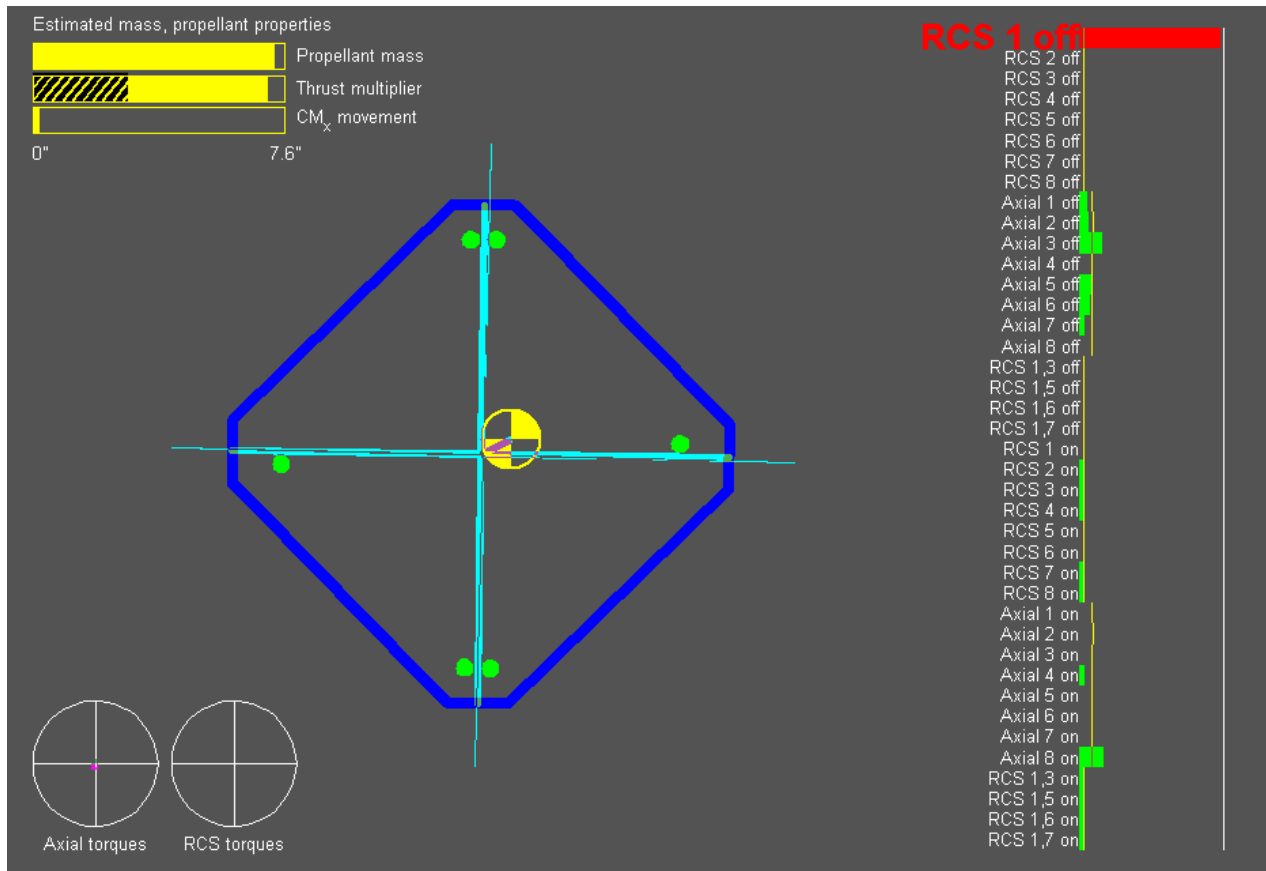
**Figure 6. Disturbing accelerations, estimated and expected**

Figure 6 shows the estimated disturbing accelerations,  $\hat{\alpha}_{disturbing}$ , for this run, along with the expected disturbing accelerations corresponding to the first fault mode,  $\hat{\alpha}_{disturbing,1}$ . Whenever jet 1 is commanded to fire (around 3, 15, and 19 seconds), there is a sharp change in the disturbing acceleration, particularly in roll. This is the information that the FDI system used to detect and isolate the fault mode. The background noise level is determined primarily by the  $\hat{\alpha}$  estimation error (driven by gyro noise), but is also affected by the pulse-to-pulse variations in thruster force. A bias in the background  $\hat{\alpha}_{disturbing}$  is caused by misalignment and bias in the thrusters (particularly the axial thrusters that are on continuously) and imperfect knowledge of the vehicle mass properties. Neither thruster-bias ID nor mass-property ID is used in computation of  $\hat{\alpha}_{disturbing}$  at this point. Note that there is an outlier on the very first sample (blue dot around  $-0.68$ ) that could have caused erroneous fault detection if this data were analyzed one point at a time rather than with the windowing approach. This data is relatively clean, and accurate detection of these faults is not as challenging as will be shown in later examples with an axial fault and with low tank pressure.



**Figure 7. FDI simulation legend**

The left side of the legend in Figure 7 refers to the thruster history plot shown in Figure 5 (and similar later figures), while the right side refers to the vehicle and FDI animation shown in Figure 8 (and similar later figures).



**Figure 8. X-38 animation display of position, FDI results, and BTI status**

The animation screen in Figure 8 was from the final update of this simulation run. The results of fuel burn-time-integration are shown in the upper left corner. The thrust (blowdown) multiplier varies from 1.0 with full tanks to about 0.38 with empty tanks, which is the reason for the cross hatched segment in the thrust multiplier bar between 0.0 and 0.38. The rear-view of the DPS is indicated, showing the attitude and that no RCS jets are firing, but 6 of the 8 axial jets are on. The likelihood monitor bars on the right side are drawn with width equal to  $e^{-0.5\lambda_{active,i}}$  so they approach 1.0 if the fault is true.  $\lambda_{active,1}$  is close to 1.0, and since RCS jet 1 has been identified as failed, it is highlighted in red. Many of the likelihood monitor bars are not drawn because insufficient information is present – for example, there is no information for Axial jet 1 failed on since it has never been commanded to not fire. The thin yellow vertical lines indicate the expected width of the likelihood monitor bars if no faults are present. The axial and RCS-torque monitors in the lower left corner would indicate torques due to these thrusters if they were commanded to fire.

Figure 9 shows  $e^{-0.5\lambda_{active,i}}$  for each fault mode, which is the metric used to graphically indicate likelihood of each fault mode (corresponds to the width of the fault mode indicator rectangles in Figure 8). A value of 0 indicates an unlikely fault mode, while a value of 1 indicates a likely fault mode. Figure 10 shows a zoomed in view of the raw likelihood parameters,  $\lambda_{active,i}$  for each fault mode. A lower value indicates a more likely fault mode (due to a better fit to the data). The blue dots correspond to RCS jet 1 failed off, and the likelihood parameter clearly approaches zero, indicating a close match, once the window is filled with data following the fault onset.

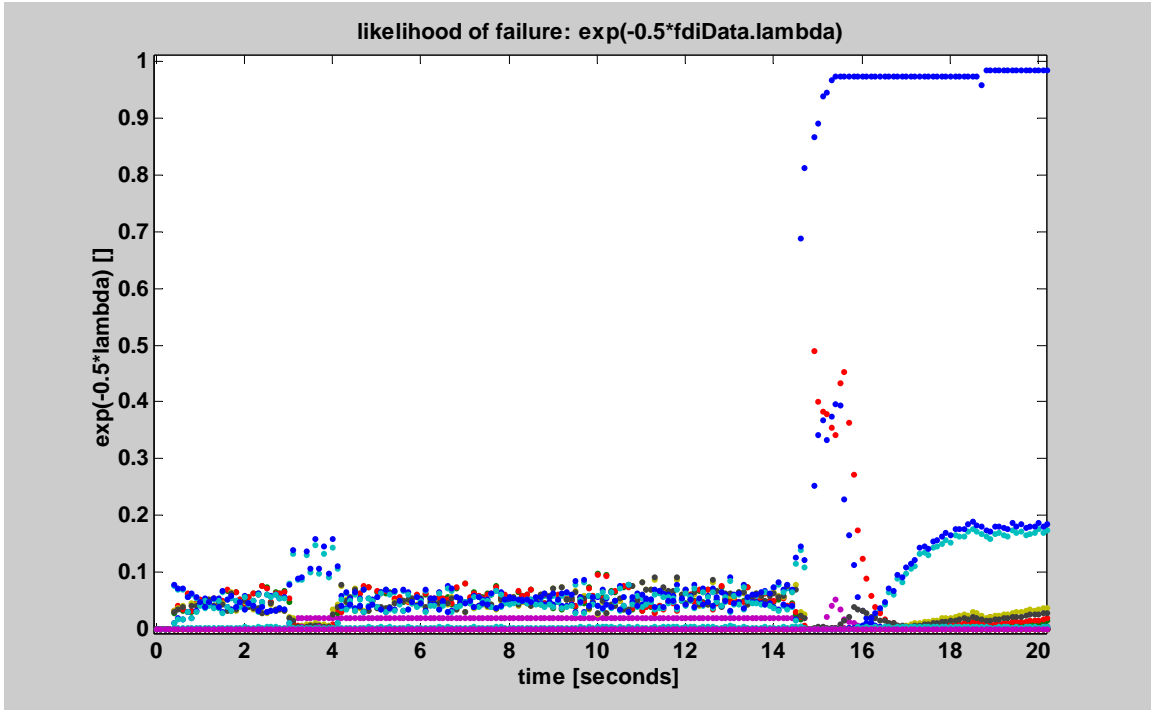


Figure 9. Likelihood indicator

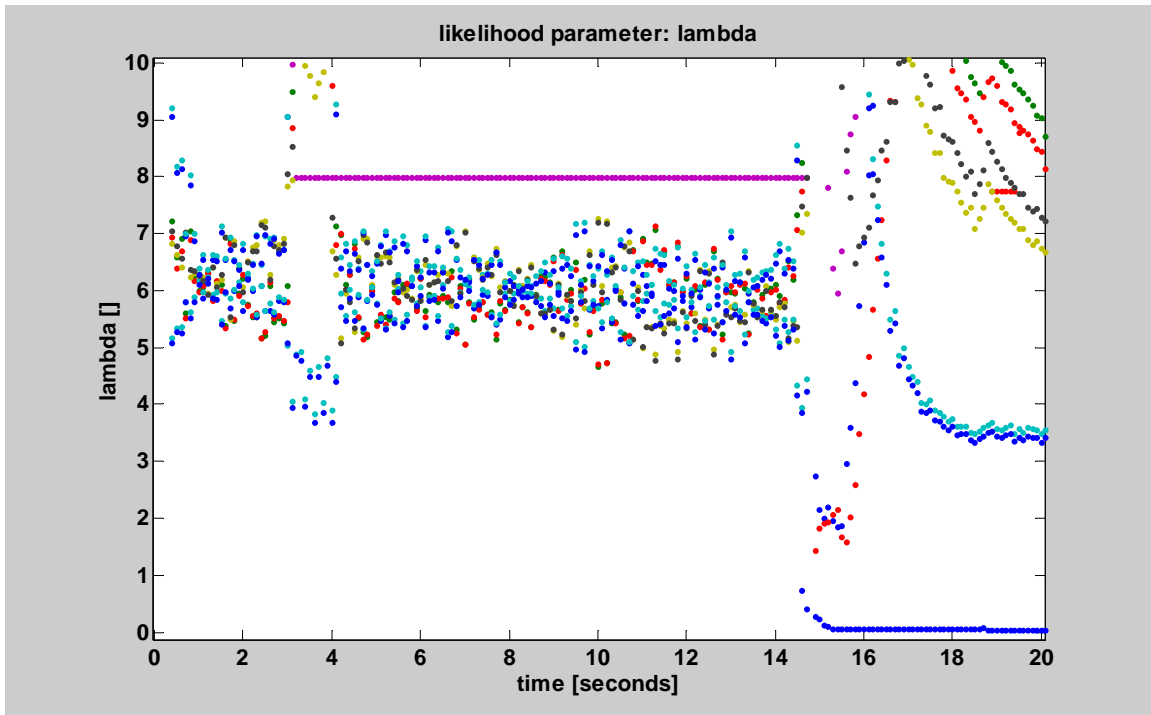


Figure 10. Lambda\_active for all fault modes, zoomed in

## 2. Axial jet failed off

Figure 11 and Figure 12 are taken from a run on the X-38 simulation where an axial thruster was failed with relatively full fuel tanks. In this case, Axial thruster 1 failed off abruptly (fault mode #9) at time = 10.0 seconds. This was detected 0.8 seconds later, indicated by the vertical red line in Figure 11. Since Axial thrusters 1 and 2 are

very close to one another and pointing in the same direction, it is virtually impossible to distinguish one from the other while their firings are synchronized – which they typically are. To address this, at time = 12.0 seconds, the axial thruster firing pattern is changed to a state that still is nominally symmetric. RCS jets were regulating to a fixed attitude as in the previous example, but they are omitted from Figure 11.

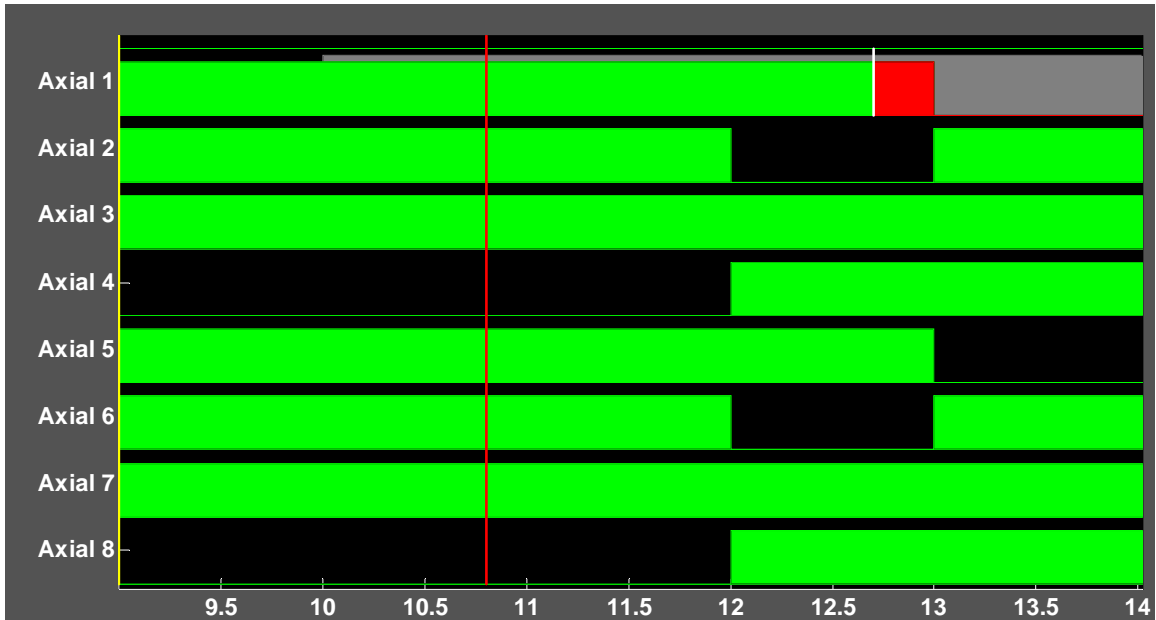


Figure 11. Thruster history, X-38 axial fault example

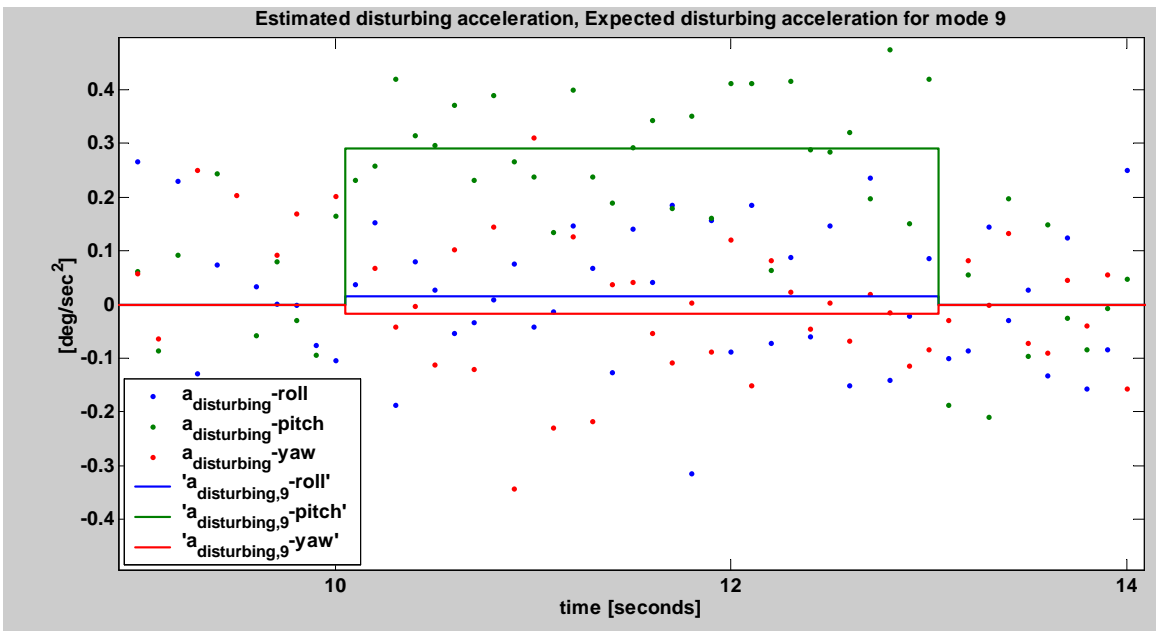


Figure 12. Estimated disturbing acceleration, X-38 axial fault example

Because of the symmetric layout and synchronized firing of the axial thrusters, fault modes 9, 10, 33, and 34 all have nearly identical disturbing acceleration signatures, as listed in Table 1\*\*\*\*.

\*\*\*\* Use of translational acceleration in this case would have easily eliminated modes 33 and 34 from consideration provided the acceleration measurement was sufficiently accurate.

Fault			expected disturbing acceleration (deg/sec <sup>2</sup> )		
mode #	thruster	type	roll	pitch	yaw
9	axial #1	off	0.0157	0.2914	-0.0164
10	axial #2	off	-0.0004	0.2914	0.0181
33	axial #5	on	0.0168	0.2941	-0.0185
34	axial #6	on	0.0006	0.2941	0.0159

**Table 1. Axial fault aliasing**

One of the main challenges of the FDI system is to quickly and accurately choose the correct choice out of these four options. This task is handled by the post-processing of the maximum likelihood calculations as follows. At the time of detection (time = 10.8 seconds in this case), it is postulated that if any fault modes have not been *active* recently, they are probably not true since they would not have been able to trigger the detection<sup>††††</sup>. So in this case, faults 33 and 34 are exonerated at time = 10.8 seconds (along with modes 2, 4, 8, 12, 16, 17, 18, 19, 20, 29, 30, 31, and 35). Then, after waiting a few samples (this delay was conservatively added to reduce false classifications), fault modes are exonerated based upon the value of the likelihood parameters  $\lambda_{active,i}$  and  $\lambda_{inactive,i}$ , provided that the calculation of those parameters is based on some minimum number of samples. Following is the display from the program while running. Note that k = 115 corresponds to time = 11.4 seconds, for example.

```

Fault detected at k = 109 (based on data through k = 108)
At time of detection, these modes were exonerated:
  2  4  8 12 16 17 18 19 20 29 30 31 33 34 35
Fault  1 exonerated at k = 115 because lambda_inactive = 16.35 > 3.03
Fault  3 exonerated at k = 115 because lambda_inactive = 16.35 > 3.03
Fault  5 exonerated at k = 115 because lambda_inactive = 16.35 > 3.03
Fault  6 exonerated at k = 115 because lambda_inactive = 16.35 > 3.03
Fault 24 exonerated at k = 116 because lambda_inactive = 6.08 > 3.03
Fault 27 exonerated at k = 116 because lambda_inactive = 6.08 > 3.03
Fault 11 exonerated at k = 117 because lambda_active = 11.89 > 1.21
Fault 13 exonerated at k = 117 because lambda_active = 22.85 > 1.21
Fault 14 exonerated at k = 117 because lambda_active = 22.69 > 1.21
Fault 15 exonerated at k = 117 because lambda_active = 11.15 > 1.21
Fault 21 exonerated at k = 117 because lambda_active = 47.10 > 3.03
Fault 22 exonerated at k = 117 because lambda_active = 51.90 > 3.03
Fault 23 exonerated at k = 117 because lambda_active = 26.80 > 3.03
Fault 25 exonerated at k = 117 because lambda_active = 34.87 > 3.03
Fault 26 exonerated at k = 117 because lambda_active = 38.01 > 3.03
Fault 28 exonerated at k = 117 because lambda_active = 28.62 > 3.03
Fault 32 exonerated at k = 117 because lambda_active = 9.80 > 1.21
Fault 36 exonerated at k = 117 because lambda_active = 13.06 > 1.21
Fault 37 exonerated at k = 120 because lambda_active = 136.40 > 3.03
Fault 38 exonerated at k = 120 because lambda_active = 137.73 > 3.03
Fault 39 exonerated at k = 120 because lambda_active = 13.83 > 3.03
Fault  7 exonerated at k = 121 because lambda_active = 34.55 > 3.03
Fault 10 exonerated at k = 127 because lambda_inactive = 8.67 > 1.21
Fault 40 exonerated at k = 128 because lambda_active = 154.76 > 3.03
Fault isolated: fault mode #9 Isolated at k = 128 based on data through k = 127
by process of elimination - all other faults were exonerated

```

At each FDI update, the fault detection logic checks all the modes that have not been exonerated to see if  $\lambda_{active,i}$  is below a certain threshold. When exactly one fault mode fits this criterion, some final checks are done and if it passes these checks, the fault mode is identified. In this particular example, all of the remaining 39 fault modes were (correctly) exonerated between time = 10.8 and time = 12.7 seconds, allowing the correct fault mode to be identified by a process of elimination. Note that in this case,  $\lambda_{active,i}$  corresponding to axial thruster #2 failed off was still very low – so it was essential to use the information contained in  $\lambda_{inactive,i}$  to remove this mode from consideration, as was done at time = 12.6 seconds.

---

<sup>††††</sup> An exception is made here for multiple-jet fault modes, since one of the sub-modes could trigger the detection while the multiple-jet mode is inactive.

Figure 12 shows the disturbing accelerations corresponding to the correct fault mode. Note how the signal-to-noise ratio (magnitude of the expected disturbing acceleration compared with the background noise level) is much lower than that for the RCS fault detection shown in Figure 6.

### 3. Axial jet failed off with tanks near empty

Figure 13 through Figure 15 are taken after a simulation run in which an axial thruster was failed off while the fuel tanks were near empty, meaning the blowdown multiplier was about 0.38 and the signal-to-noise ratio was significantly reduced. To account for this reduced SNR, the sizes of the filtering windows are automatically increased accordingly to permit the averaging of more data.

In this case, axial jet #3 was failed off at time = 5.0 seconds, this was detected at time = 7.2 seconds, active toggling of the axial thrusters began at time = 10.0 seconds, and the correct fault mode was identified at time = 12.1 seconds. In this case (as with the previous example) after fault isolation, the axial thrusters are toggled to a state that leaves the fault *inactive*. (i.e., a stuck-off thruster is commanded to be off, and a stuck-on thruster would be commanded to be on).

Figure 13 shows the animation of the vehicle at the end of the run. The empty fuel tanks and corresponding effect on thrust multiplier and CM location is displayed in the upper left corner. The maximum likelihood indicator rectangles on the right indicate that Axial 3 off, Axial 4 off, and RCS 1 and 6 off (modes 11, 12, and 19) all have very low values of  $\lambda_{active,i}$  (indicated by the width of the rectangles being close to 1.0). In this case, the two alias fault mode candidates (12 and 19) could not be removed from consideration based on direct evaluation of the  $\lambda_{active,i}$  and  $\lambda_{inactive,i}$  values, so the decision logic dropped into a more complex mode.

In this mode, the  $\lambda_{active,i}$ ,  $\lambda_{inactive,i}$ , and  $\lambda_{ratio,i}$ <sup>\*\*\*\*</sup> are checked for each candidate mode. For these modes, the corresponding values are listed in Table 2.

Fault			calculated FDI variable		
mode #	thruster	type	$\lambda_{active,i}$	$\lambda_{inactive,i}$	$\lambda_{ratio,i}$
11	axial #3	off	0.0065	0.0162	0.0054
12	axial #4	off	0.2704	1.8616	0.9116
19	RCS #1,#6	off	0.5769	0.5586	0.9142

**Table 2. Axial and multi-jet RCS fault aliasing**

In this logic, some more rigorous tests are done, and unless they are all passed, no fault is identified. Only the mode with the smallest  $\lambda_{active,i}$ , (mode 11 in this example), is considered, and both  $\lambda_{active,11}$  and  $\lambda_{ratio,11}$  must be below their corresponding thresholds. Then  $\lambda_{inactive,i}$  is evaluated, which can be complicated since frequently not enough data is available to calculate a value (a self-imposed limitation – a minimum number of relevant samples must be available before this calculation is made). When available,  $\lambda_{inactive,11}$  must be below the corresponding threshold, and must also be below  $\lambda_{inactive,i}$  for the mode with the second-smallest  $\lambda_{active,i}$  (mode 12 in this example). As the final check, the ratio between the smallest and second-smallest  $\lambda_{active,i}$  must be below a threshold. This level of complexity in the logic was required only to achieve the very high success rates for the X-38

---

<sup>\*\*\*\*</sup> As presented in Section III.F,  $\lambda_{active,i0}$  uses data recorded at the same times as  $\lambda_{active,i}$ , but compares to an expected disturbing acceleration of 0.  $\lambda_{active,i0}$  should be close to zero if the fault is false, and  $\lambda_{active,i}$  should be close to zero if the fault is true. Both values increase with the relative noise level. The ratio,  $\lambda_{ratio,i}$ , should then be close to zero if the fault is true, much greater than 1 if the fault is false, and close to 1 if the signal is very low compared to the noise.

application, and was entered only in very low SNR cases like this. For the other vehicles, and for the X-38 with full tanks, accurate FDI could be achieved simply by tuning the few thresholds for  $\lambda_{active,i}$  and  $\lambda_{inactive,i}$ .

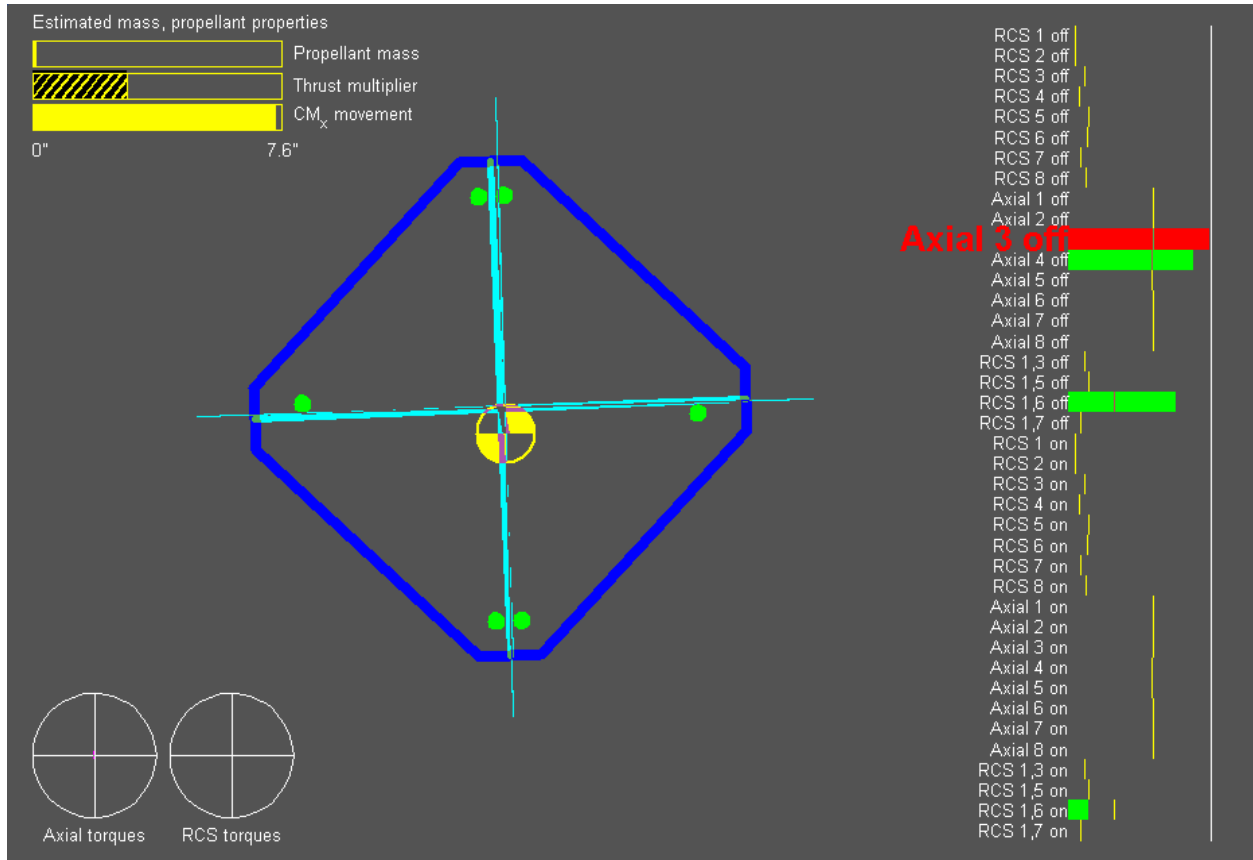


Figure 13. Animation display, X-38 axial fault with empty tanks

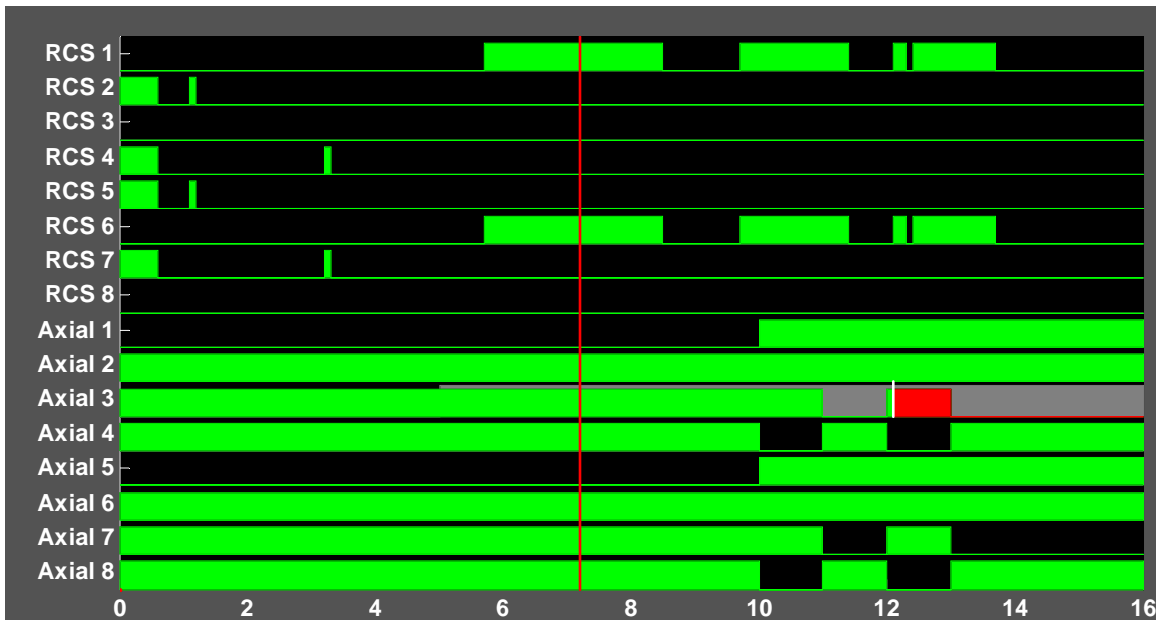


Figure 14. Thruster command history, X-38 axial fault with empty tanks

Figure 14 shows the thruster command history for this run. Note that the times required to detect and isolate the faults are longer than for the run with full tanks, shown in Figure 11. This is because the required minimum sizes of the filtering windows were increased to account for the reduced signal to noise ratio.

Figure 15 shows the estimated disturbing acceleration, along with the expected disturbing acceleration for the true fault mode, 11. The SNR is clearly very low in this case, and successfully isolating the fault mode is very challenging. This is a clear example of what necessitated the development of the full FDI system as presented here, in contrast to the simpler maximum-likelihood core as was developed by Deyst and Deckert<sup>7</sup>.

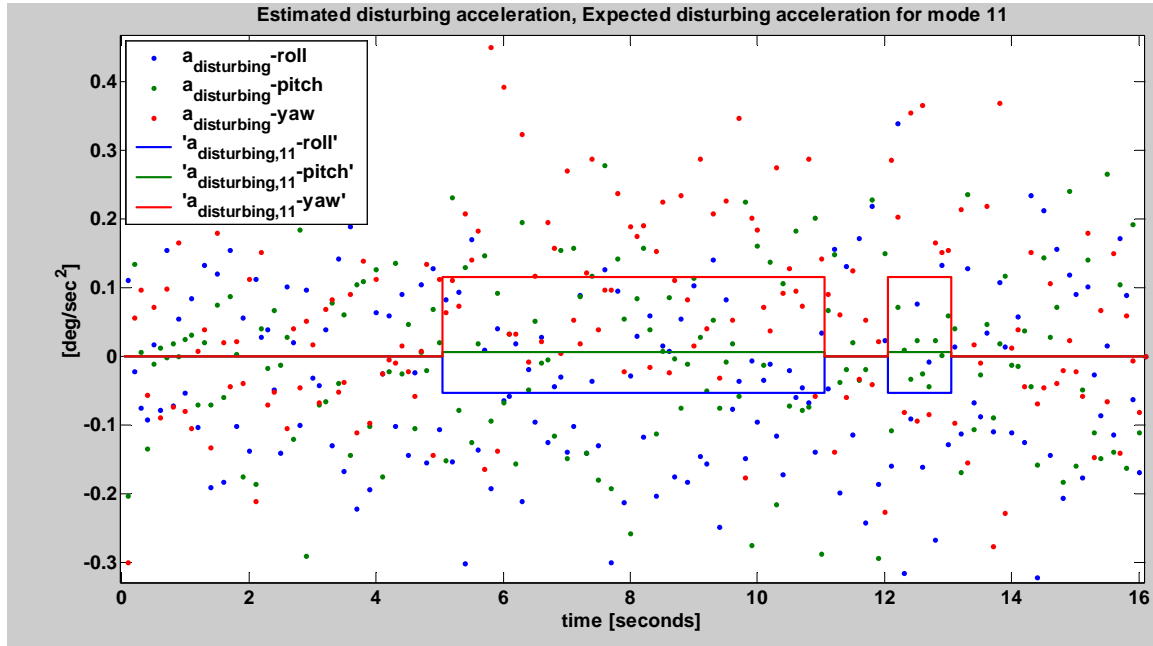


Figure 15. Estimated disturbing acceleration, X-38 axial fault with empty tanks

## B. Bank of Kalman Filters

Figure 16 through Figure 21 demonstrate the bank-of-Kalman-filters FDI approach. The thruster history is shown in Figure 16, and smoothed residuals from 4 members of the Kalman-filter bank are shown in Figure 17 through Figure 21. The smoothed results were generated by passing the raw residuals (much noisier) through a forward-backward single pole filter, reducing the noise without introducing a phase lag. As this filtering approach is acausal, some modification to this would be required before implementing the bank of Kalman filters and using it for real-time FDI.

The true fault mode in this test was Axial jet 3 failed off, which is fault mode #11.

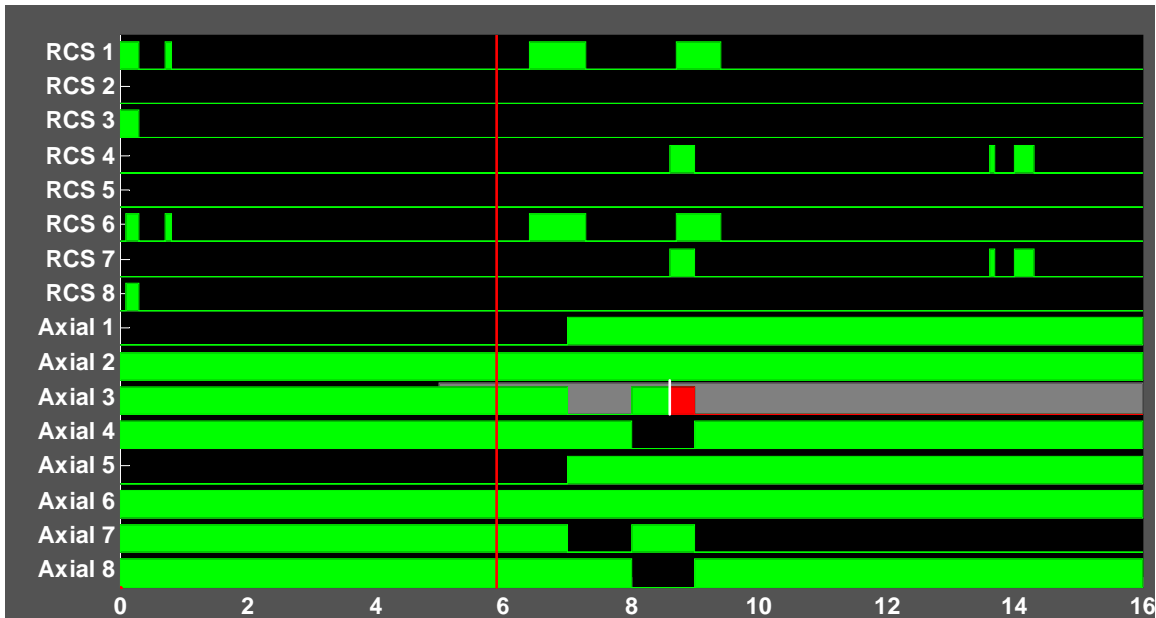


Figure 16. Thruster command history, Bank of KFs example

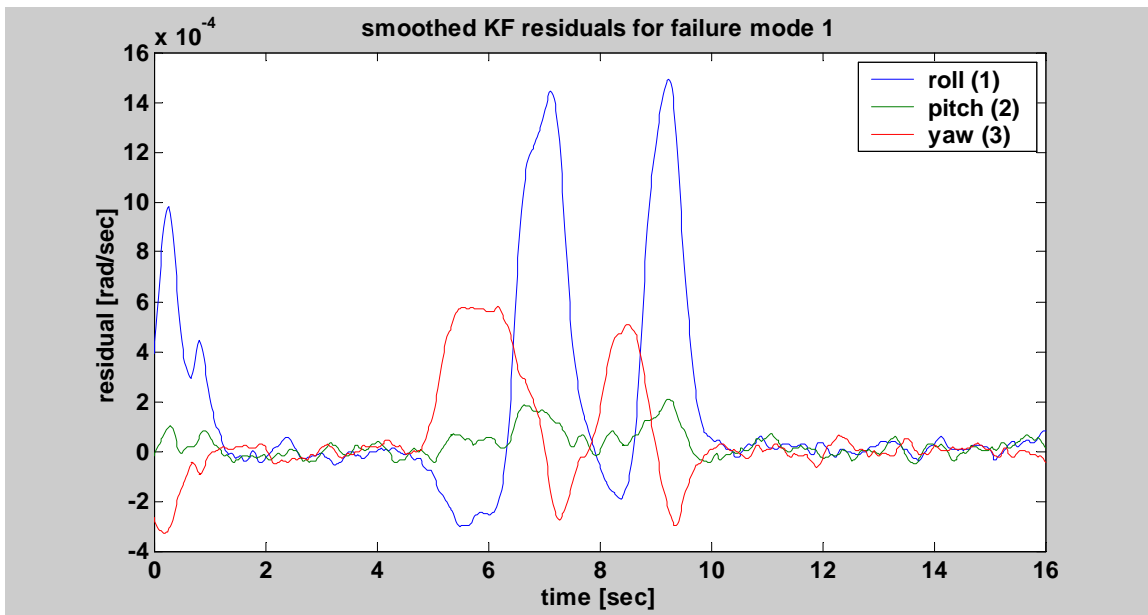
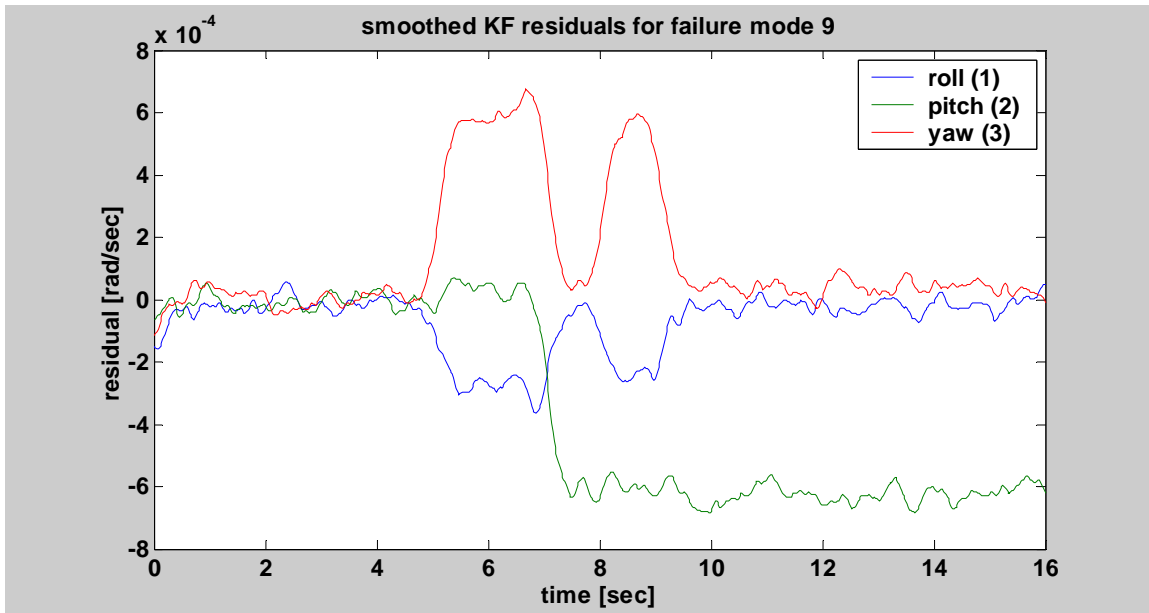


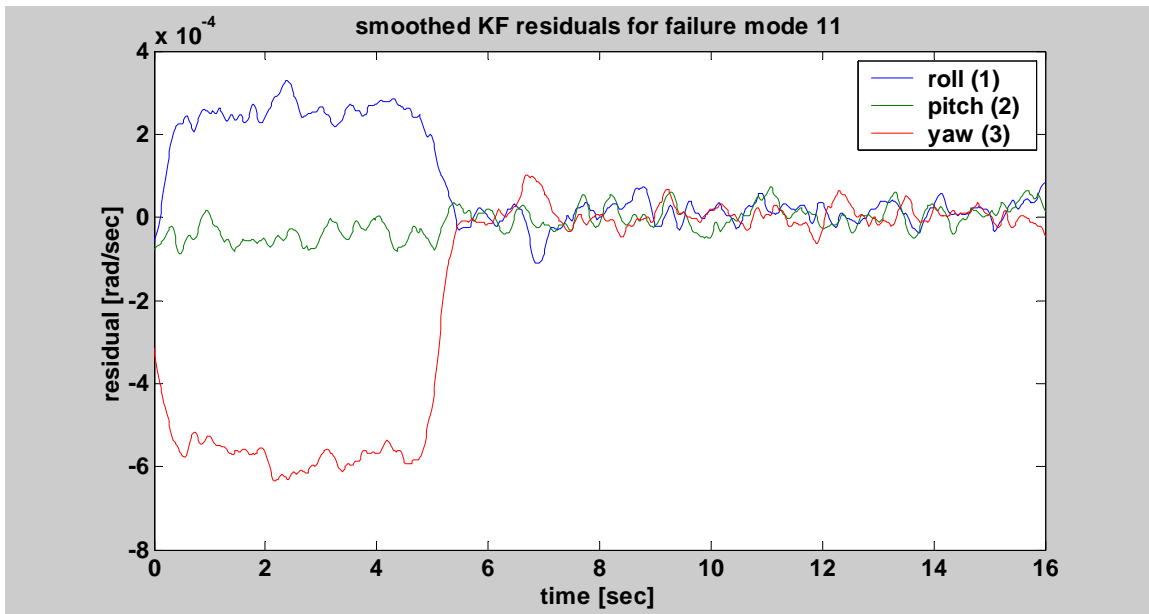
Figure 17. Smoothed residuals for KF with RCS jet 1 failed off

The KF whose residuals are shown in Figure 17 uses a system model that includes RCS jet 1 failed in the off-position. Since this is not true, residual spikes are seen at times when RCS jet 1 fires – these are positive spikes in roll and negative in yaw, occurring around 0-1, 7, and 9 seconds. There are also significant residuals at the times when Axial thruster 3 is (truly) failed and is commanded to fire, occurring around 5-7 and 8-9 seconds. This behavior is exactly as expected, and the spikes that occur at times when RCS jet 1 fires as well as when Axial jet 3 fires after it has failed would indicate that this fault mode is not true.



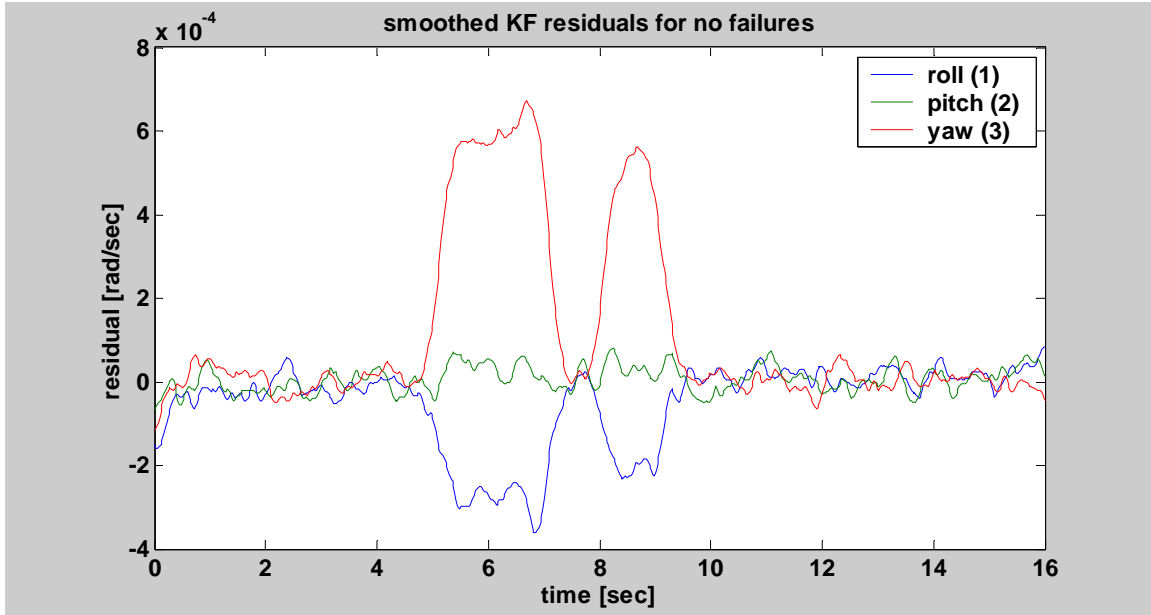
**Figure 18. Smoothed residuals for KF with axial jet 1 failed off**

Figure 18 shows similar results for Axial jet 1 failed off, except that due to the continuous nature of the axial thruster control, the residuals look more like biases than spikes. These biases occur (in different directions) when the true fault is *active* or when axial jet 1 is commanded to fire.



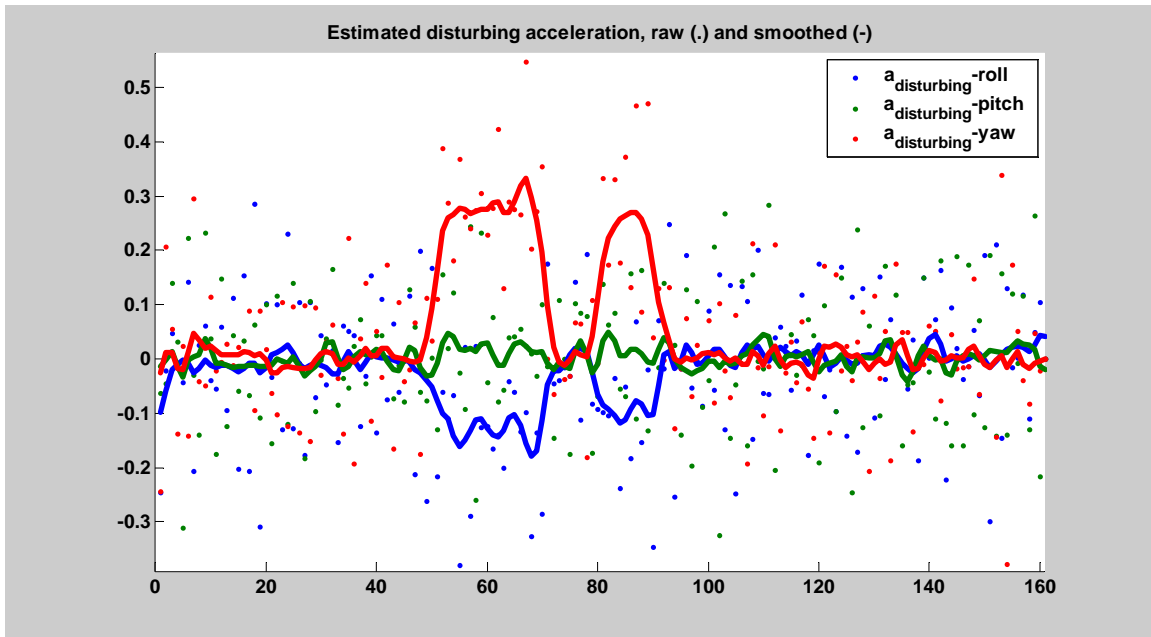
**Figure 19. Smoothed residuals for KF with axial jet 3 failed off (true fault)**

Figure 19 shows the residuals for the KF corresponding to the true fault. The residuals start out high while the fault mode is not present, but then at time=5.0 seconds, the residuals drop quickly to zero when the fault appears abruptly at that time. It then stays near zero from then on, regardless of whether the true fault mode is active or which other thrusters fire. This appears to show the potential of using this as an FDI tool, but it would be more complex in the case of an RCS jet fault, which is generally pulsed.



**Figure 20. Smoothed residuals for KF with no faults**

Figure 20 shows the residuals for a KF that assumes no faults are present. This clearly responds when the true fault mode is *active*, from 5-7 and 8-9 seconds. However, the KF approach did not appear to offer a clear improvement over the maximum-likelihood approach, so it was not pursued further.



**Figure 21. Smoothed estimated disturbing acceleration**

For comparison, Figure 21 shows the smoothed disturbing acceleration, which looks very similar to the KF residuals. This is surprising that they are so similar, since they are derived using very different approaches (although starting with the same raw data source – the gyros). Also note that the units in these two plots are different since one is an acceleration estimate while the other is the residual from a velocity estimate. The similarity between these two plots supports the decision to stick with the maximum-likelihood approach, as it highlights the fact that the KF does not add any information that is not already available in the disturbing acceleration estimate, and would be more difficult to work with when dealing with the windowing, since the data has been filtered.

### C. Mini AERCam thruster FDI

In this section, a typical thruster FDI simulation run is presented for the Mini AERCam, very similar to the X-38 results in Section IV.A. In this 60-second simulation, the vehicle was commanded to regulate both position and attitude, starting from an initial state error, and then limit cycling. Thruster #1 was failed off at time = 4.0 seconds. Sensors were read and the EOM were integrated at 100 Hz, and the thruster commands were updated at 25 Hz. Since many aspects of this simulation are similar to the X-38 simulation discussed in Section IV.A, only new aspects will be discussed.

Figure 22 and Figure 23 show the thruster command history. The pulses (each lasting 2 control time steps, or 0.08 seconds) at around time = 13, 27, and 42 seconds all contribute to the FDI even though the fault is not detected until time = 52.72 seconds and identified at time = 53.16 seconds as shown in Figure 23. The total (firing) time the fault is active is 0.40 seconds before fault detection and 0.36 more seconds before isolation.

Each thruster provides both a force and a torque on the vehicle. So when jet 1 was commanded to fire at time = 13, 27, and 42 seconds, the required torque was provided by jet 7 that was also fired at those times. However, the fault resulted in a net force in the +x body direction, eventually causing the controller to request a translational correction around time = 52 seconds. When a force and torque such as this are requested simultaneously (-x translation and +z rotation) firing thruster 1 will produce the desired force and torque directions (at half the full strength). For that reason, the thruster stays commanded on from that point on, expediting the FDI. Eventually, the vehicle will rotate enough so that the x-axis (inertial frame) can be controlled using the 3-5-9-11 set of thrusters.

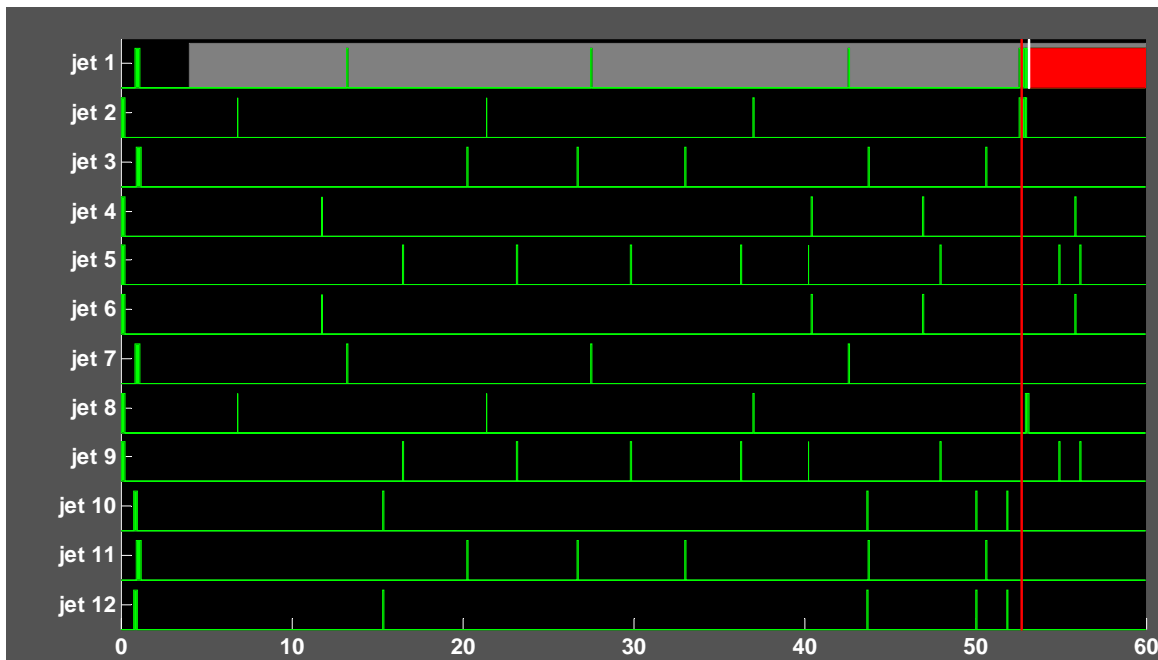


Figure 22. Thruster command history, FDI results

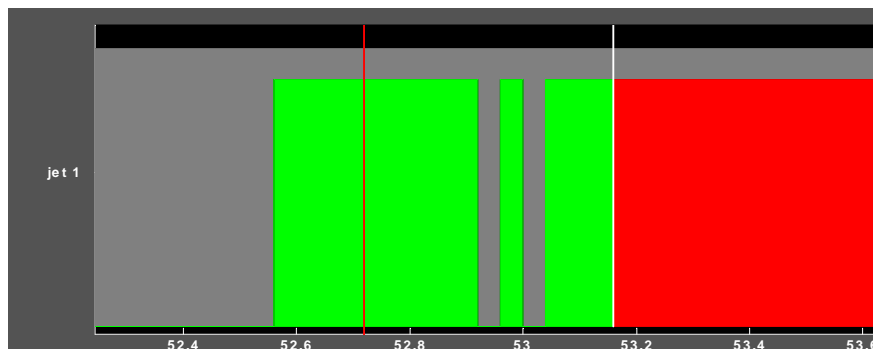


Figure 23. Thruster command history, FDI results, zoomed in

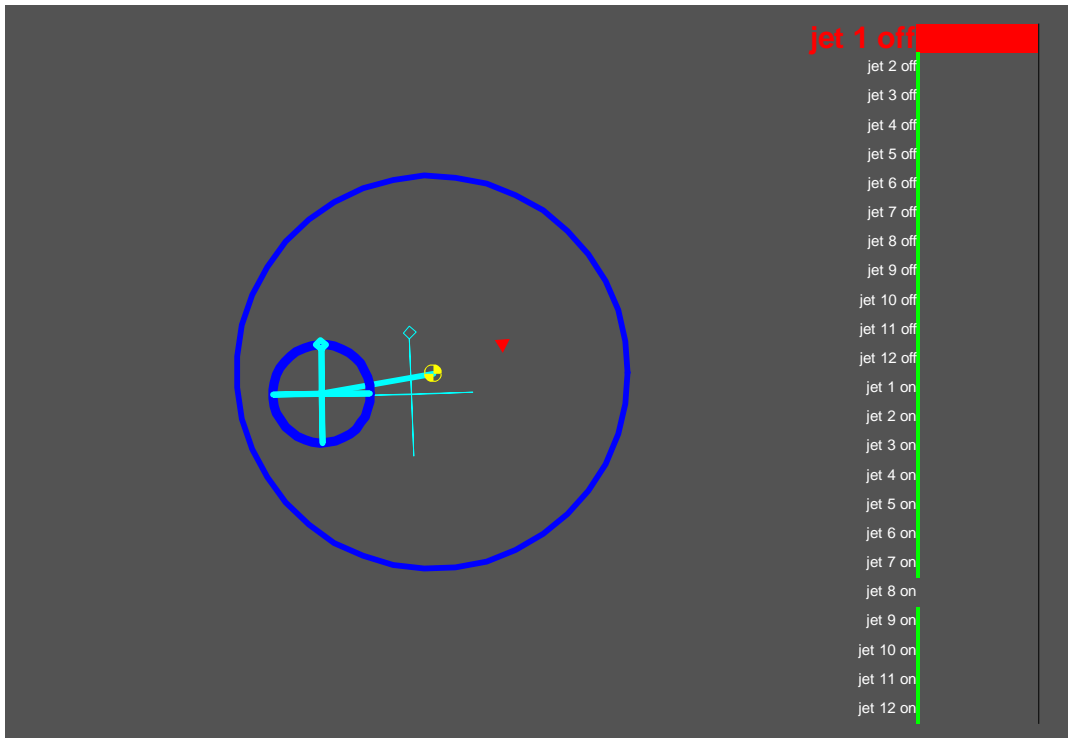


Figure 24. Mini AERCAM animation display

Figure 24 shows the Mini AERCAM animation. The deviation between desired and actual attitude is large here, since the system is not strictly controllable in the presence of a single jet fault, and no reconfiguration has been implemented. Jet 1 is on at the end of the simulation, indicated by the red triangle in the figure. The FDI indicator bars on the right show the very clear distinction between fault modes, in contrast to Figure 13, for example.

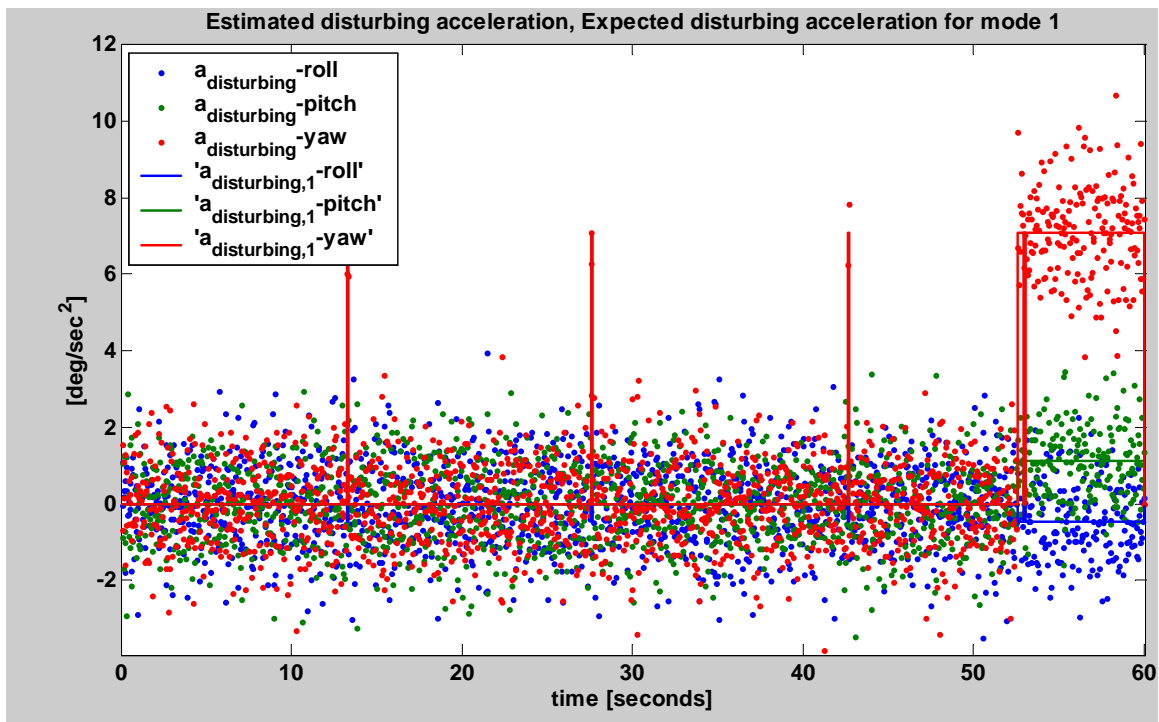


Figure 25. Disturbing accelerations, estimated and expected

Figure 25 shows the estimated and disturbing accelerations. Even though the signal-to-noise ratio here may not look much better than for the X-38, this problem is significantly easier because the expected disturbing accelerations are in different directions for most fault modes (other than some clear aliases, such as jet 1 off and jet 8 on – those are distinguished from one another by active excitation). Much of the complex decision logic developed for the X-38 (as described in Section IV.A.3) is not needed for the Mini AERCam.

Although not tested yet, it is likely that the Mini AERCam FDI can be implemented following the streamlined implementation used for SPHERES.

## V. SPHERES Implementation

The SPHERES represents the hardware platform used to validate this technology. MATLAB simulations similar to those presented in Section IV were developed for SPHERES and used throughout the real-time code development to ensure accuracy of the algorithm implementation. The TMS320C6701 floating point digital signal processor (DSP) on the SPHERES experimental spacecraft has high throughput (up to 1 Gflops), but little ROM is available on the Sundance SMT375 board, limiting the memory size for the software. This fact alone makes it infeasible to implement the full algorithm with windowing. Fortunately, the acceleration estimation is very accurate, and the fault signatures are sufficiently dispersed that the non-windowing version of the algorithm could be implemented.

The FDI algorithm has been successfully implemented in Embedded C++. The performance of the real-time code has been verified to match results of the original MATLAB code within precision expected with floating point processing. Testing on the air-bearing table shown in Figure 4 has demonstrated highly accurate and robust thruster FDI. Some details of the steps in this streamlined algorithm, as they differ from the full algorithm are presented below.

At startup:

- 24 thruster accelerations (also fault signatures) are pre-calculated – a 6x24 matrix (6 includes the 3 rotational and 3 translational accelerations, 24 includes 12 off fault modes and 12 on). This is done in MATLAB, using calibrated parameters for the SPHERES being used, and MATLAB code automatically generates the C code containing this matrix. This facilitates switching between different SPHERES and the automation reduces the chances of coding or other manual errors.

During each control period:

- The measured acceleration vector is estimated.
- Multi-jet thrust scale factors are calculated, based on the number of thrusters commanded to fire, then plus/minus one in case an on/off fault is active.
- Disturbing acceleration is calculated from measured and nominal (with no faults) accelerations.
- Likelihood parameters (lambdas) are calculated for each of the 12 active modes plus the case of no faults, using the 6x24 matrix, disturbing acceleration, and accounting for the multi-jet scale factors
- Disturbance detector check – At least one of the 13 likelihood parameters must be below a threshold (a close match). If the lowest one is above threshold, it means that the disturbing acceleration did not match any of the “possible” thruster fault conditions, and the sample is thrown out. This situation will occur when the spacecraft is handled by the Astronaut, bumps a wall, or for the air-bearing testing – the air bearing grounds out on some dust. This very simple check was implemented to prevent the experiment from failing due to Astronaut handling, which may be necessary due to the limited workspace on the ISS.
- Detection is based on a generalized likelihood ratio test, just as in the full algorithm.
- Upon detection, a list of fault candidates is initialized as the 12 that were active during the detection cycle.
- This list of 12 can generally immediately (still working with the measurement from the detection control cycle) be whittled down to 4 or 2 by exonerating those with high likelihood parameters (a poor fit), either when *active* or *inactive*. The 2 or 4 remaining are aliases of one another. If accelerometers are used it will be 2, if not, usually 4.
- A very efficiently implemented automatic excitation algorithm finalizes isolation in one or two more cycles, with all but one candidate exonerated based on lambda *active* or *inactive*.
- The isolation is made based on process of elimination – all but one fault will have been exonerated.
- If the fault is an on-fault it is over-ridden and turned off permanently.
- A simple reconfigured controller is switched in. Since the SPHERES uses the minimal set of 12 thrusters to control 6 degrees of freedom (DOF), when one is lost, the spacecraft is underactuated, strict controllability is lost, and a nonholonomic controller is required to control all 6 DOF. The simple

controller used here does not attempt to control attitude on the axis controlled by the lost thruster. Position regulation is maintained in 3 axes, and attitude is controlled in 2 out of the 3 axes.

The SPHERES will be launched with a supply of consumables: CO2 tanks and battery packs. The experiments have been designed to optimize experimental value constrained by limited gas, power-on time (limited by battery life), and total experiment time (limited by astronaut time).

The SPHERES program is designed to run over a sequence of 10 or more 2-hour experiment sessions, space two or more weeks apart. Each 2-hour session will typically contain experiments from multiple guest scientist teams (mass ID, FDI, formation flying, docking, etc.). The results from one session will be analyzed on the ground, and possibly refined and extended in subsequent sessions.

1. The first experiment session will run mass-property ID tests, with the results of those tests being used to determine the model parameters<sup>3</sup>.
2. The second session contains the initial thruster FDI experiments
  - a. Closed loop position and attitude control
  - b. Controlled to follow sequence of way-points
  - c. Initiate thruster-off fault in software, followed by automatic FDI and simple R
  - d. Repeat with different thrusters, on and off faults
  - e. Thruster commands, on-board-filtered rate and angular acceleration, and raw gyro signals (as permitted by limited communications bandwidth) will be telemetered from all experiments to permit off-line testing and analysis.
3. Subsequent sessions:
  - a. Multiple faults
  - b. More advanced control reconfiguration
  - c. Docking following thruster fault

An early iteration of the mass ID algorithms was tested in a series of NASA KC-135A (“Vomit Comet”) flights from November 5-8, 2003, providing zero-g testing to complement the extensive 1-g air-bearing table testing performed at MIT and NASA Ames. These results were presented in Ref. 18, and along with the 1-g testing, confirm that the acceleration estimation accuracy and thrust repeatability is sufficient for robust FDI with the non-windowing algorithm.

## VI. Conclusion

A maximum-likelihood-based thruster FDI algorithm has been developed and applied in simulation to the X-38 Mini AERCAM, and MIT SPHERES thruster-controlled vehicles. Using gyro signals only, or with the additional capability provided with accelerometers, the algorithm reliably and accurately detects and isolates hard, abrupt single- and multiple-jet on- or off-faults in the presence of several significant noise sources. The algorithm is computationally efficient and scales better than linearly with the number of failure modes to be identified. Since the algorithm makes use of pre-existing navigation sensors, this fault-tolerant capability could be provided as a software-only addition to a new spacecraft or modification to an existing one.

A simplified version of the algorithm has been implemented and tested on the MIT SPHERES experimental spacecraft. Zero-g flight testing of the acceleration estimation algorithms aboard the NASA zero-g KC-135A aircraft has demonstrated the viability of this streamlined algorithm. Testing of the SPHERES on an air-bearing table has demonstrated reliable thruster FDI and a rudimentary control reconfiguration. An experimental plan for upcoming tests aboard the ISS has been presented.

## Appendix

### A. Abbreviations

CM:	Center of mass
CO2:	Carbon dioxide
DOF:	Degree(s) of freedom
EOM:	Equation(s) of motion
FDI:	Fault detection and isolation
FDIR:	Fault detection isolation and reconfiguration (or recovery)
ID:	Identification (system identification)
ISS:	International Space Station

LS:	Least squares
MCRLS:	Multiple concurrent recursive least squares
Mini AERCam:	Miniature autonomous extravehicular robotic camera
NASA:	National Aeronautics and Space Administration
RF:	Radio frequency
RLS:	Recursive least squares
ROM:	Read only memory
RWA:	Reaction wheel assembly
SNR:	Signal to noise ratio
SPHERES:	Synchronized position hold, engage, reorient, experimental satellites
US:	Ultrasound

### Acknowledgments

Research funded by NASA Headquarters, HQ AA, PWC 349-00, the NASA Intelligent Systems Program 302-10-10 (part of the CICT Program), and a NASA STTR award to Payload Systems, Inc. Contract #NNA05AC52C. Thanks to Dustin Berkovitz, Dr. Edmund Kong, Prof. David Miller, Simon Nolet, Steve Sell, and the rest of the MIT / Payload Systems, Inc. SPHERES team for assistance with the SPHERES integration. Thanks to Rodolfo Gonzalez and Dr. Steven Fredrickson of NASA Johnson Space Center for assistance in the problem formulations for the X-38 and Mini AERCam applications.

### References

- <sup>1</sup>Wilson, E., Lages, C.R., and Mah, R.W., "Gyro-Based Maximum-Likelihood Thruster Fault Detection and Identification," *2002 American Control Conference*, Anchorage, AK, May 2002.
- <sup>2</sup>Wilson, E., Lages, C.R., and Mah, R.W., "On-line, Gyro-Based, Mass-Property Identification for Thruster-Controlled Spacecraft," *2002 IEEE International Midwest Symposium on Circuits and Systems*, Tulsa, OK, Aug. 2002.
- <sup>3</sup>Wilson, E., Sutter, D.W., and Mah, R.W., "Motion-Based Mass- and Thruster-Property Identification for Thruster-Controlled Spacecraft," *AIAA Infotech@Aerospace Conference*, Arlington, VA, Sep. 2005.
- <sup>4</sup>Isermann, R., "Process Fault Detection Based on Modeling and Estimation Methods – A Survey," *Automatica*, Vol. 20, No. 4, 1984, pp. 387-404.
- <sup>5</sup>Deyst, J. J. and Deckert, J. C., "Maximum likelihood failure detection techniques applied to the Shuttle RCS jets," *Journal of Spacecraft and Rockets*, Vol. 13, No. 2, February 1976, pp. 65-74.
- <sup>6</sup>Wilson, E. and Rock, S.M., "Reconfigurable Control of a Free-Flying Space Robot Using Neural Networks," *1995 American Control Conference*, Seattle, WA, Jun. 1995.
- <sup>7</sup>Wilson, E., *Experiments in Neural Network Control of a Free-Flying Space Robot*, Ph.D. Dissertation, Mechanical Engineering Dept., Stanford University, Stanford, CA 94305, SUDAAR 666, Mar. 1995.
- <sup>8</sup>Lee, A.Y. and Brown, M.J. "Model-based thruster leakage monitor for the Cassini spacecraft," *Journal of Spacecraft and Rockets*, Vol. 36, No. 5, 1999, pp. 745-749.
- <sup>9</sup>Willms, B., "Space integrated GPS/INS (SIGI) navigation system for Space Shuttle," *Digital Avionics Systems Conference*, 1999.
- <sup>10</sup>Wagenknecht, J. D., and González, R. A., "Guidance, Navigation, and Control (GN&C) for the Autonomous Extravehicular Robotic Camera (AERCam) Free-Flyer Vehicles," *AIAA ISS Service Vehicles Conference*, Apr. 1999.
- <sup>11</sup>Wagenknecht, J.D., Fredrickson, S., Manning, T., and Jones, B., "Design, Development and Testing of the Miniature Autonomous Extravehicular Robotic Camera (Mini AERCam) Guidance, Navigation, and Control System," *26th Annual American Astronautical Society Guidance and Control Conference*, Feb. 2003.
- <sup>12</sup>Saenz-Otero, A., Miller, D.W., "The SPHERES ISS Laboratory for Rendezvous and Formation Flight", *ESA Guidance, Navigation, and Controls Conference*, #29, 2002.
- <sup>13</sup>Nolet, S., Kong, E., and Miller, D.W., "Autonomous Docking Algorithm Development and Experimentation Using the SPHERES Testbed," *SPIE Defense and Security Symposium*, Orlando, FL, Apr. 2004.
- <sup>14</sup>Othon, W. and Strack, D. *X-38 Flight Dynamics Team Note of Interest: Reaction Control System and Blowdown Tank Design Document*, Note #245, Mar. 1999.
- <sup>15</sup>Gelb, A, et al., *Applied Optimal Estimation*, MIT Press, Cambridge, MA, 1974.
- <sup>16</sup>Van Trees, H.L., *Detection, Estimation, and Modulation Theory, Part I*, Wiley, New York, 1968.
- <sup>17</sup>MATLAB is a registered trademark of The MathWorks, Inc., 24 Prime Park Way, Natick, MA 01760, 508-647-7000.
- <sup>18</sup>Wilson, E., Sutter, D.W., et al., "Motion-Based System Identification and Fault Detection and Isolation Technologies for Thruster Controlled Spacecraft," *JANNAF 3rd Modeling and Simulation Joint Subcommittee Meeting*, Colorado Springs, CO, Dec. 2003.