NASA CONTRACTOR REPORT 178252

A DUAL-PROCESSOR MULTI-FREQUENCY IMPLEMENTATION OF THE FINDS ALGORITHM

```
(NASA-CR-178252)  A DUAL-PROCESSOR                          N87-23617
MULTI-FREQUENCY IMPLEMENTATION OF THE FINDS
ALGORITHM Final Report  (Charles River
Analytics)  59 p  Avail: NTIS HC  A04/MF                    Unclas
A01                                    CSCL 01D G3/06       0077638
```

PANKAJ M. GODIWALA AND ALPER K. CAGLAYAN

CHARLES RIVER ANALYTICS INC.
CAMBRIDGE, MA 02138

CONTRACT NAS1-17719
APRIL 1987

NASA
National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665-5225

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

In this report, we present the progress made in NASA Contract NAS1-17719, entitled "Evaluation of Fault Tolerant Concepts," since the supplemental Final Report of March 1986 [1]. The main result of this phase of the project is the partitioning of the FINDS algorithm into two parts: one processing the sensors related to the translational dynamics and the other processing the sensors related to the rotational kinematics, thus allowing a parallel processing solution on the dual-processor configuration of the target flight computer. These partitioned sub-algorithms have been ported onto the two sides of the target flight computer, using a DMA local data communication link to transfer the relevant program variables between the two sides. The functional performance and execution speed of this parallel implementation has been evaluated on the flight computer using five minutes of flight recorded sensor data [2], [6] from the NASA ATOPS B-737 aircraft in a Microwave Landing System (MLS) environment.

As specified in [1], the target flight computer chosen for this phase of the project has a dual-processor configuration with each side having 128 Kb of memory. Each processor has approximately a 255,000 Whetstones floating point performance in 32 bit single precision. In the previous phase, the FINDS algorithm software size had been reduced to 116 Kb allowing the entire composite algorithm (combined translational and rotational) to be ported onto only one side of the target flight computer. The executable image of this composite version ran about 11/3.5 times slower than real time at gain update frequencies of 20/4 Hz, using only a single side CPU. The decision to retain this same target flight computer in the current phase of the project thus necessitated the split-up of the algorithm to allow a real-time parallel processing solution.

The major modifications made to the composite FINDS algorithm in the current phase are as follows:

(a) No-Fail Filter (NFF) Auto-Initialization

In all of the test runs reported in [1], the no-fail filter (NFF) which is an extended Kalman filter (EKF), had been assigned initial state conditions (viz. aircraft position, velocity, attitudes and horizontal winds) based on preliminary hand calculations performed with the flight data. Thus, an automatic filter initialization routine was developed by using the first iteration of the flight data to calculate the NFF initial condition state estimates.

(b) Isolation Strategy Changes

As discussed in Chapter 3 in [1], we had replaced the FINDS multiple hypothesis test performing simultaneous detection and isolation with a hierarchical detection and isolation test performing isolation only in the event of a failure detection. In [1], only the failure detection strategy had been reported. Here, we present the corresponding sensor failure isolation algorithm compatible with the new detection strategy.

(c) Partitioning of the FINDS Algorithm

As mentioned earlier, the FINDS algorithm was partitioned into two sub-algorithms to fully utilize the dual configured target flight computer in a parallel execution mode. This split-up was possible without any impact on the NFF estimation performance since the translational dynamics filter had already been decoupled from the rotational kinematics in the previous phase by using a constant state transition matrix. In the partitioned FINDS algorithm, there are two no-fail filters: one for the rotational kinematics and the other for the translational dynamics.

The rotational kinematics filter uses the IMU attitude and rate gyro sensors in a 3-state, 3-bias configuration. The FINDS software for the rotational kinematics has an executable image of 74 Kb and runs at a speed of 1.35 times real time on the target flight computer at the nominal 20 Hz gain update frequency.

The translational dynamics filter uses the MLS and IAS sensors along with the linear accelerometers, resulting in a 8-state and 3-bias filter configuration. The FINDS software for the translational dynamics has an executable size of 87 Kb and executes at approximately 3.55 times real time on the target flight computer at the nominal 20 Hz frequency.

The aircraft state and sensor bias estimation performance of both subsets has been tested and is identical to the baseline performance of the composite FINDS algorithm [1]. However, the FDI performance of both subsets improves marginally as the failure detection thresholds are now dependent on the NFF residuals involving only dynamically correlated sensors on each side. This results in a slightly lower failure detection time for some of the test cases reported in [1].

(d) Multi-Rate Implementation

Based on a study of the low gain update frequency performance of the FINDS algorithm, we have investigated the use of different update frequencies for the bias-free and bias filter computations in the NFF. Further analysis also indicated that the gains can be updated at lower frequencies than the covariances with very little degradation in estimation performance from nominal results. Based on this analysis, we have made the necessary modifications allowing a multi-rate implementation of the bias-free and bias filters. In this multi-rate implementation of the FINDS algorithm, bias-free gain, bias-free covariance, bias filter gain, and bias filter covariance matrices are updated at different frequencies. This allows for various combinations of bias-free/bias filter and gain/covariance update frequencies. A few of the combinations which yield an execution (CPU usage) time lower than the simulated flight time, with acceptable estimation and FDI performance, are presented here.

Chapter 2 presents the modifications (a) and (b) above along with the isolation performance results for the FINDS algorithm using the hierarchical detection and isolation test. In Chapter 3, we discuss in detail the changes (c) and (d) above and review the impact of these modifications on the estimation and FDI performance. Chapter 4 concludes the report with our final remarks and recommendations for algorithmic modifications and performance evaluations.

II. NFF INITIALIZATION AND FINDS FAILURE ISOLATION PERFORMANCE

In this chapter, we discuss the various changes made in the FINDS algorithm after the new failure detection strategy had been implemented [1]. In particular, the NFF was modified by incorporating an auto-initialization routine which sets up the initial state estimates based on the first iteration of the flight data -- this routine is discussed in the first section. The next section deals with the modifications in the failure isolation strategy to make it compatible with the new hierarchical detection and isolation test. The modified isolation strategy now takes into account the different moving window lengths used by the detection algorithm.

2.1 NFF Auto-Initialization routine

In order to avoid large initial error transients, the Extended Kalman Filter (EKF) used as the no-fail state estimator in the FINDS algorithm has to be initialized as close to the true states as possible. In [1], [2], and [6] all the test runs were executed by initializing the filter states using preliminary hand calculations with the flight data. For flight testing the FINDS algorithm, we have developed and tested a filter initialization routine. This routine reads the first iteration of the flight data and uses the MLS azimuth, elevation, and range, IAS, IMU roll, pitch and yaw measurements to calculate the NFF states which consist of the aircraft position, velocity, attitude and horizontal winds in the runway frame.

Figure 2.1 depicts the MLS geometry and measurements with reference to the MLS and runway coordinate systems. Let x_{az}, y_{az}, and z_{az} be the x, y and z coordinates of the azimuth/range antenna with respect to the runway frame. Similarly, let x_{el}, y_{el}, and z_{el} be the x, y and z coordinates of the elevation antenna in the runway coordinate system. Then, $x_{oe} = x_{az} - x_{el}$,

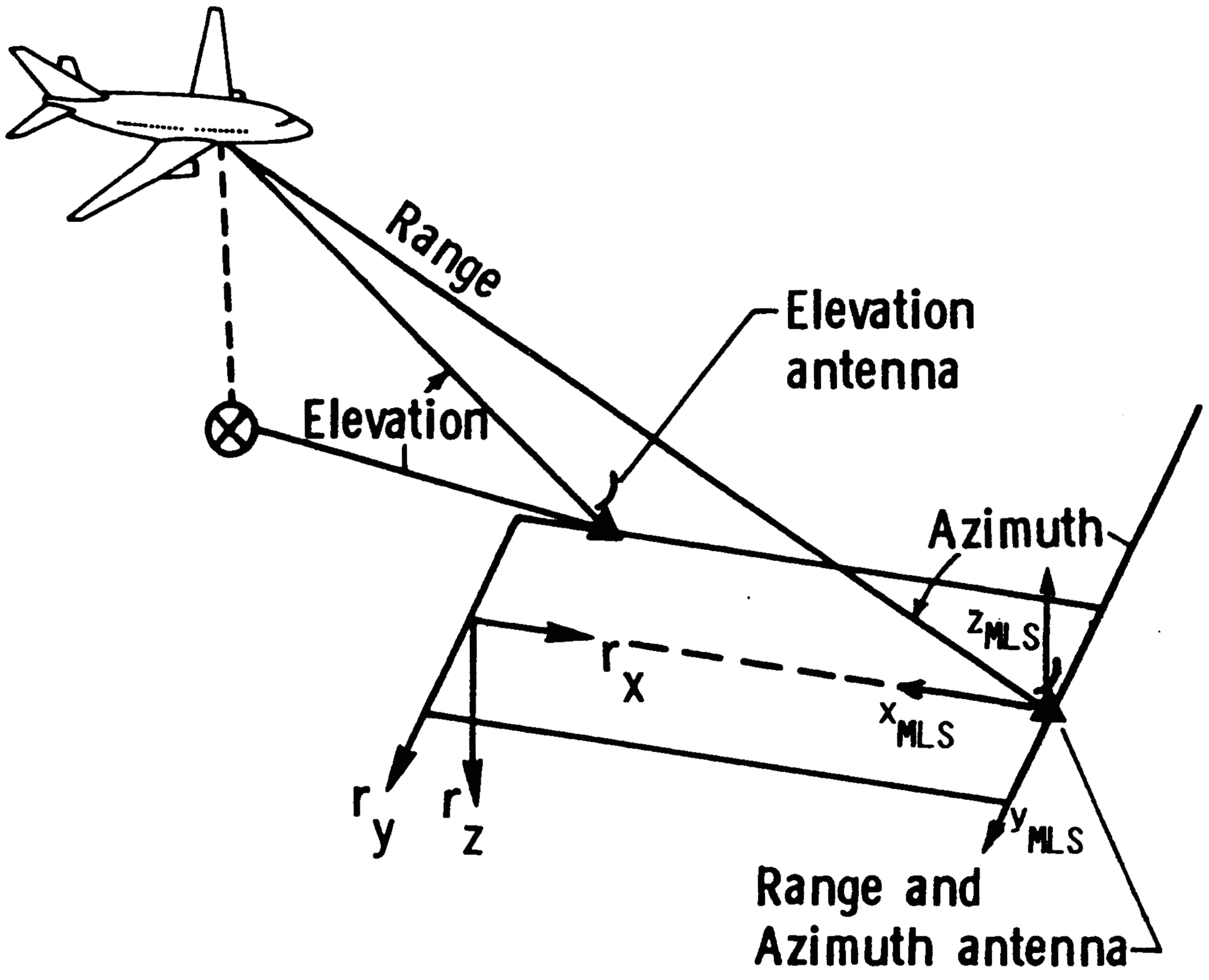


Figure 2.1: Runway Coordinate System and MLS Geometry

$y_{oe} = y_{az} + y_{el}$ and $z_{oe} = z_{az} - z_{el}$ represent the coordinates of the elevation antenna with respect to the azimuth origin; i.e., in the MLS coordinate system.

If x_M, y_M, z_M denote the aircraft position in the MLS frame, then the MLS measurements are given by:

$$\text{RANGE} = (x_M^2 + y_M^2 + z_M^2)^{1/2}$$

$$\text{AZIM} = \sin^{-1}\left(\frac{-y_M}{\text{RANGE}}\right) \tag{2.1}$$

$$\text{ELEV} = \sin^{-1} \frac{(z_M - z_{oe})}{((x_M - x_{oe})^2 + (y_M - y_{oe})^2 + (z_M - z_{oe})^2)^{1/2}}$$

Inverting (2.1), we have the solution for x_M, y_M, z_M as

$$x_M = f + (f^2 - h)^{1/2}$$

$$y_M = -\text{RANGE} \cdot [\sin(\text{AZIM})] \tag{2.2}$$

$$z_M = (\text{RANGE}^2 - x_M^2 - y_M^2)^{1/2}$$

where

$$f = x_{oe} \cdot [\sin(\text{ELEV})]^2$$

$$h = (x_{oe}^2 + y_{oe}^2) \cdot [\sin(\text{ELEV})]^2 + y_M^2 - \text{RANGE}^2 \cdot [\cos(\text{ELEV})]^2$$
$$- 2 \cdot y_M \cdot y_{oe} \cdot [\sin(\text{ELEV})]^2 - (z_{oe}^2 - 2 \cdot z_M \cdot z_{oe}) \cdot [\cos(\text{ELEV})]^2$$

The term containing z_{oe} and z_M in the expression for h vanishes when there is no vertical offset between the azimuth and elevation antennae, i.e., when $z_{oe} = 0$; (2.2) then becomes a closed form solution for x_M, y_M, and z_M. When z_{oe} is not zero, (2.2) has to be solved in iterative fashion (between x_M and z_M) to reconstruct the aircraft position in the MLS frame. However, in the filter initialization routine, no significant errors are introduced by ignoring this last term and using the closed form solution of (2.2). Finally, in the runway inertial system,

$$\begin{aligned} x_{rw} &= x_{az} - x_M \\ y_{rw} &= y_{az} + y_M \qquad (2.3) \\ z_{rw} &= z_{az} - z_M \end{aligned}$$

where x_{rw}, y_{rw} and z_{rw} give us the initial aircraft position estimates in the runway frame.

The IMU roll, pitch and yaw measurements are replicated; averaging these dual channel values directly gives us the initial estimates for the aircraft attitude as

$$\begin{aligned} \phi &= (\phi_{m1} + \phi_{m2})/2 \\ \theta &= (\theta_{m1} + \theta_{m2})/2 \qquad (2.4) \\ \psi &= (\psi_{m1} + \psi_{m2})/2 - \psi_R \end{aligned}$$

where ψ_R is the runway yaw, fixed for the given runway configuration.

The generalized expressions for the aircraft velocity in the body frame are

$$\begin{aligned} v_{bx} &= \text{IAS}\cdot\cos\alpha\cdot\cos\beta \\ v_{by} &= \text{IAS}\cdot\cos\alpha\cdot\sin\beta \qquad (2.5) \\ v_{bz} &= \text{IAS}\cdot\sin\alpha \end{aligned}$$

where α is the angle of attack and β is the sideslip angle.

Given (2.5) and the Euler angles ϕ, θ and ψ from (2.4), we can transform the aircraft velocity in the body frame to the navigation frame (i.e., runway frame) and compensate for the horizontal winds, w_x and w_y, to yield

$$v_{rx} = (c\theta c\psi)\cdot IAS\cdot(c\alpha c\beta) + (s\phi s\theta c\psi - c\phi s\psi)\cdot IAS\cdot(c\alpha s\beta)$$
$$+ (c\phi s\theta c\psi + s\phi s\psi)\cdot IAS\cdot(s\alpha) + w_x$$
$$v_{ry} = (c\theta s\psi)\cdot IAS\cdot(c\alpha c\beta) + (s\phi s\theta s\psi + c\phi c\psi)\cdot IAS\cdot(c\alpha s\beta)$$
$$+ (c\phi s\theta s\psi - s\phi c\psi)\cdot IAS\cdot(s\alpha) + w_y \qquad (2.6)$$
$$v_{rz} = (-s\theta)\cdot IAS\cdot(c\alpha c\beta) + (s\phi c\theta)\cdot IAS\cdot(c\alpha s\beta)$$
$$+ (c\phi c\theta)\cdot IAS\cdot(s\alpha) + w_z$$

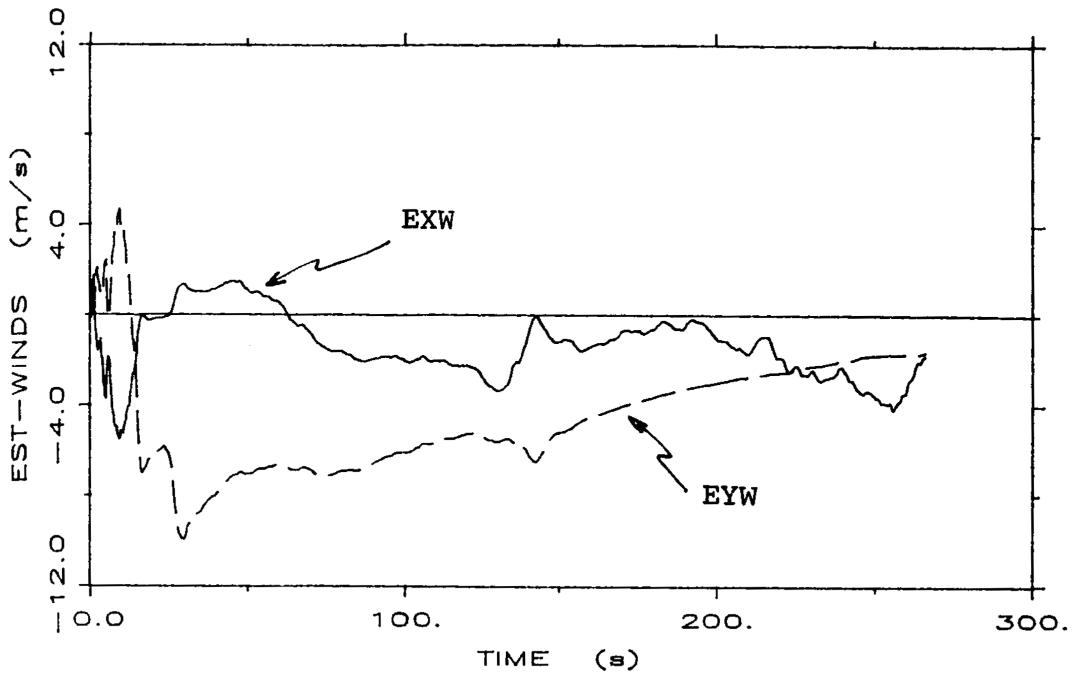
Realistically, α and β are fairly small angles and can be assumed to be zero since these expressions are used only in the filter initialization routine; and so the errors caused by this assumption are relatively insignificant and are offset by the initial uncertainty in the filter. Thus, assuming $\alpha = \beta = 0$, we get the initial estimates for aircraft velocity as

$$v_{rx} = IAS\cdot c\theta\cdot c\psi + w_x$$
$$v_{ry} = IAS\cdot c\theta\cdot s\psi + w_y \qquad (2.7)$$
$$v_{rz} = -IAS\cdot s\theta$$

The only remaining no-fail filter states to be initialized are the horizontal winds. For our initialization routine, we have decided to set both wind estimates to a value of zero. This initial estimate, combined with a higher initial uncertainty on the wind estimation error covariances (10 m/s as opposed to 0.75 m/s) and the slowly time-varying wind model [1], gave us satisfactory results in terms of wind estimation performance. In a practical situation (e.g., transition from VORTAC to MLS), the knowledge of the winds from the enroute portion of the flight can be used to initialize the wind estimates in this filter initialization scheme.

We now present a brief overview of the NFF performance with the initialization routine described above, particularly with respect to the changes in the estimation performance over that discussed in chapter 2 in [1]. The flight recorded sensor data used to evaluate this performance (and to generate all the results presented in this report) covers about 266 seconds of flight, at a sampling frequency of 20 Hz for each sensor. The flight starts when the aircraft enters MLS coverage and ends at an altitude of 30.48 m when radar altimeter activation occurs. In the runway frame, the aircraft is initially at approximately (-17000 m, -4800 m), at an altitude of 760 m and with the ground track oriented roughly 30° relative to the runway. A bank maneuver is executed from 110-150 seconds to align the aircraft with the runway. Except for a constant altitude during the runway alignment maneuver, the aircraft is descending at a steady 4 m/s sink rate. Plots for the aircraft ground track, altitude, velocity and attitude profiles have not been presented here since the NFF estimation performance for these variables is essentially the same as in figures 2.1-2.4 in [1]. Figure 2.2 shows the new horizontal wind estimates and the normal operating bias estimate for the longitudinal accelerometer. In comparison with figure 2.5 in [1], we see that the wind estimates, starting with an initial estimate of zero, converge to the true values in about 25 seconds. This is due to the high initial uncertainty placed on the initial wind estimates and the time required for the reduction in the accelerometer bias estimation uncertainty. Figure 2.3 shows the NFF bias estimates for the lateral and vertical accelerometers. For each of the accelerometers, we see that the bias estimates show larger transients when compared with earlier results (figures 2.6 and 2.7 in [1]), but the convergence time to steady state and the steady state values themselves are the same as before.

Figures 2.4 and 2.5 present the no-fail filter residual time histories for MLS azimuth, MLS elevation, MLS range and IAS, respectively. Again, the only significant change in the behavior of these residuals from those depicted in [1] (figures 2.9, 2.10) can be seen in the initial 10-15 second time span



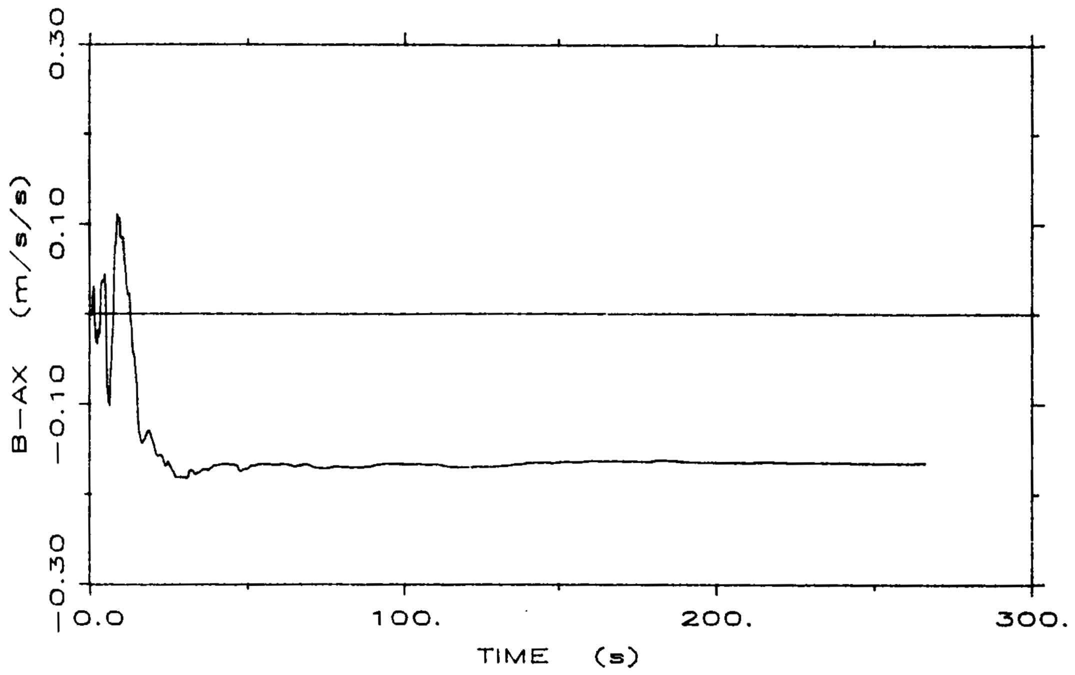
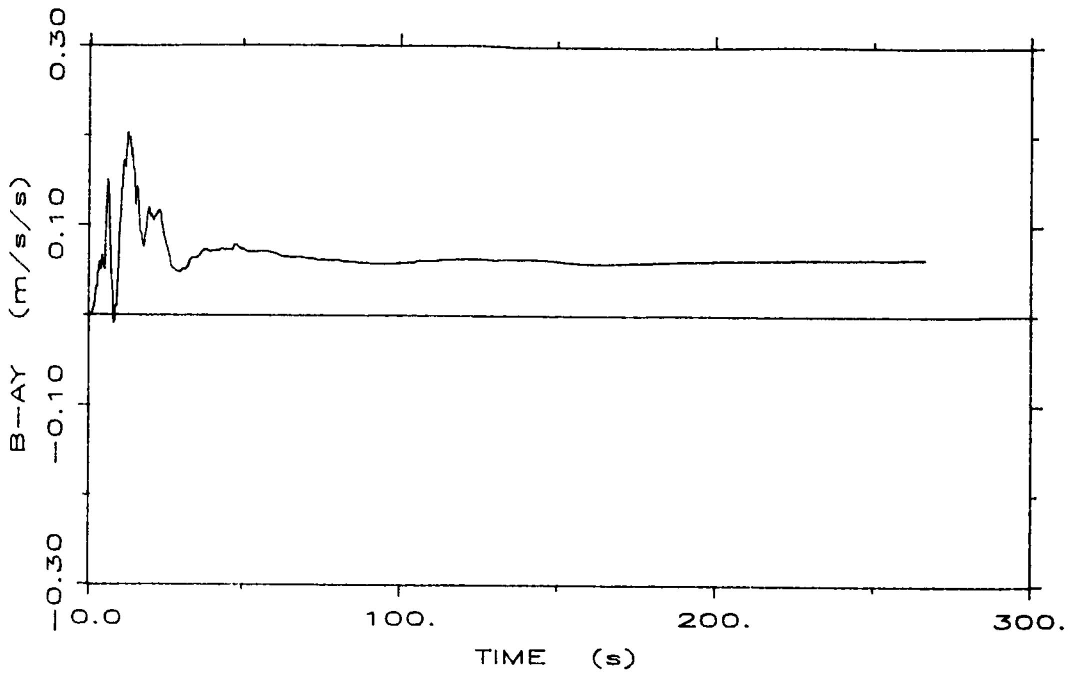


Figure 2.2: Horizontal wind and longitudinal accelerometer bias estimates



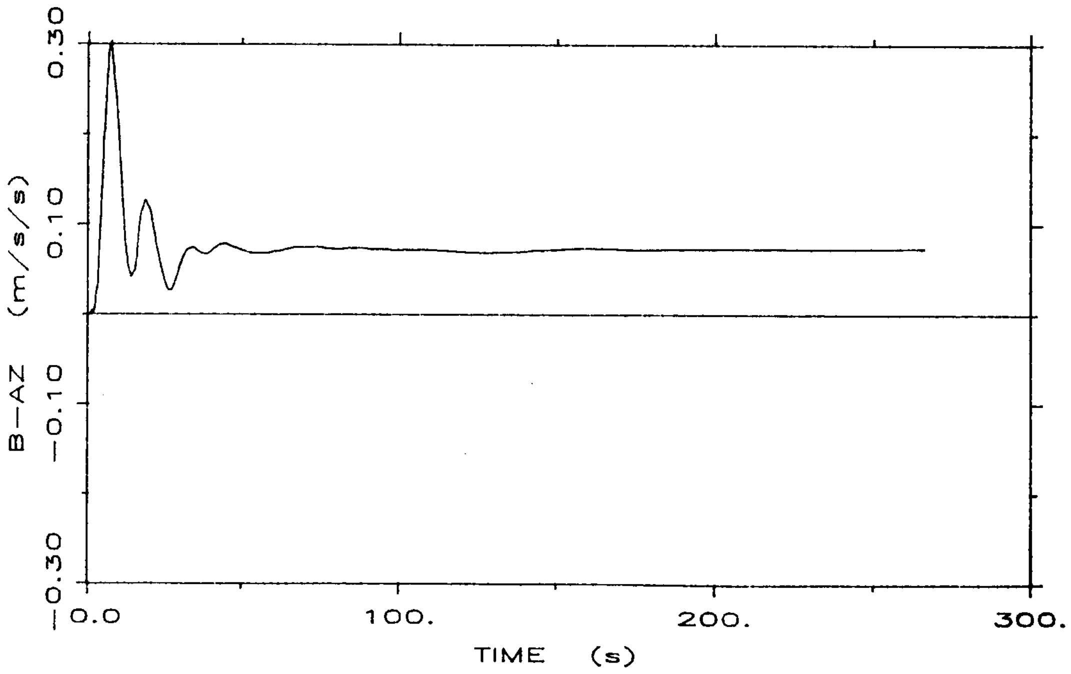
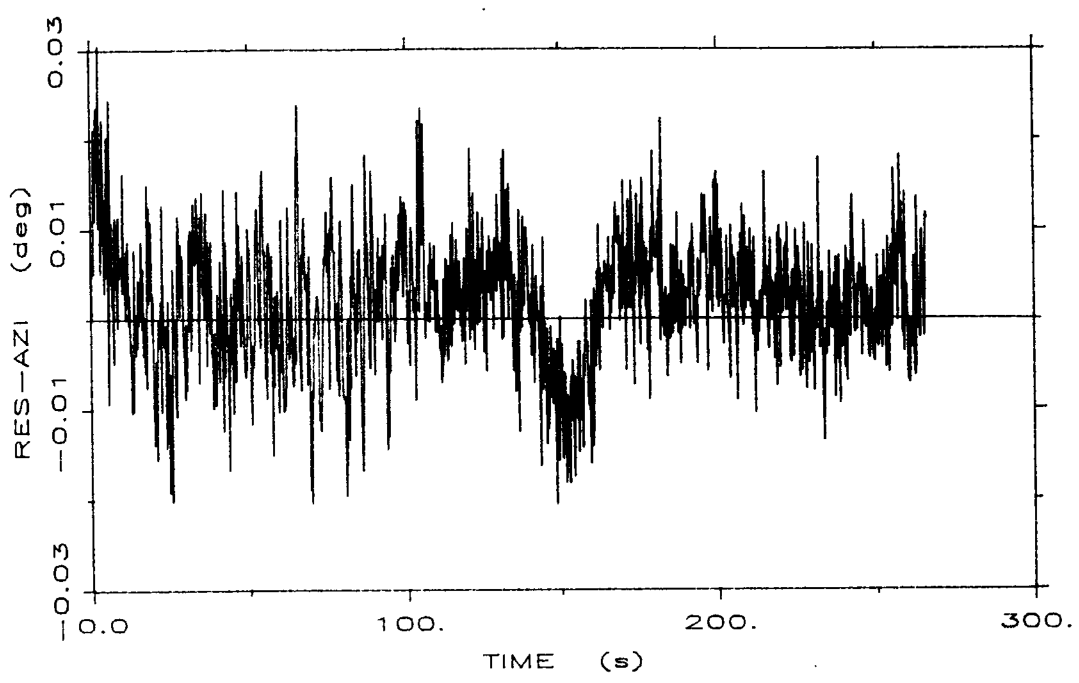


Figure 2.3: Lateral and vertical accelerometer bias estimates



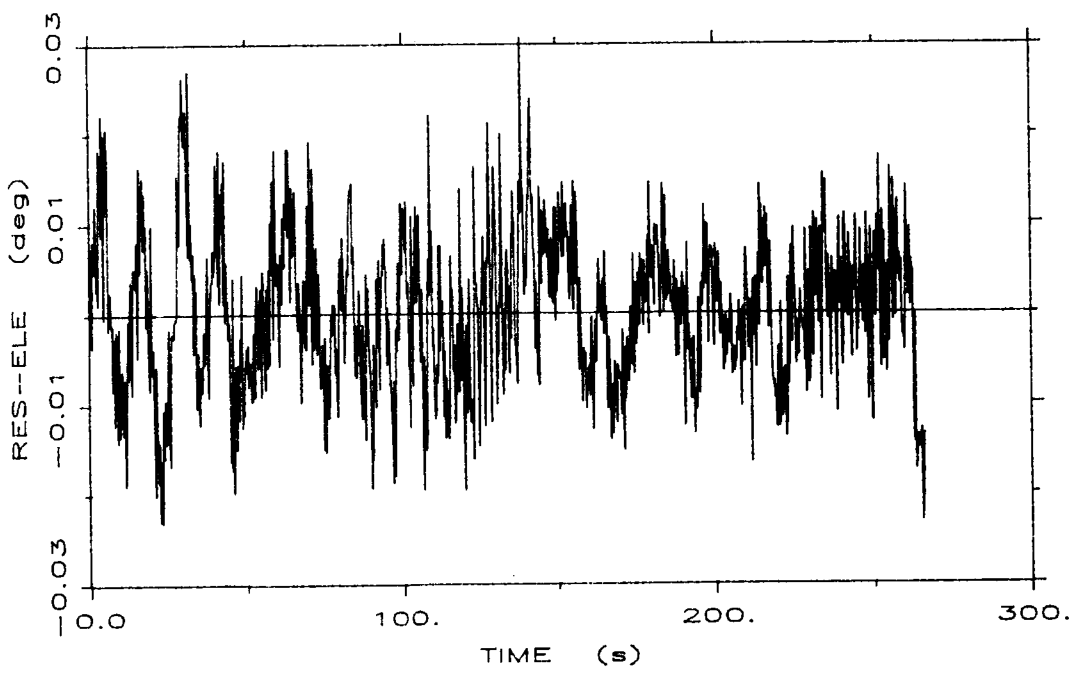
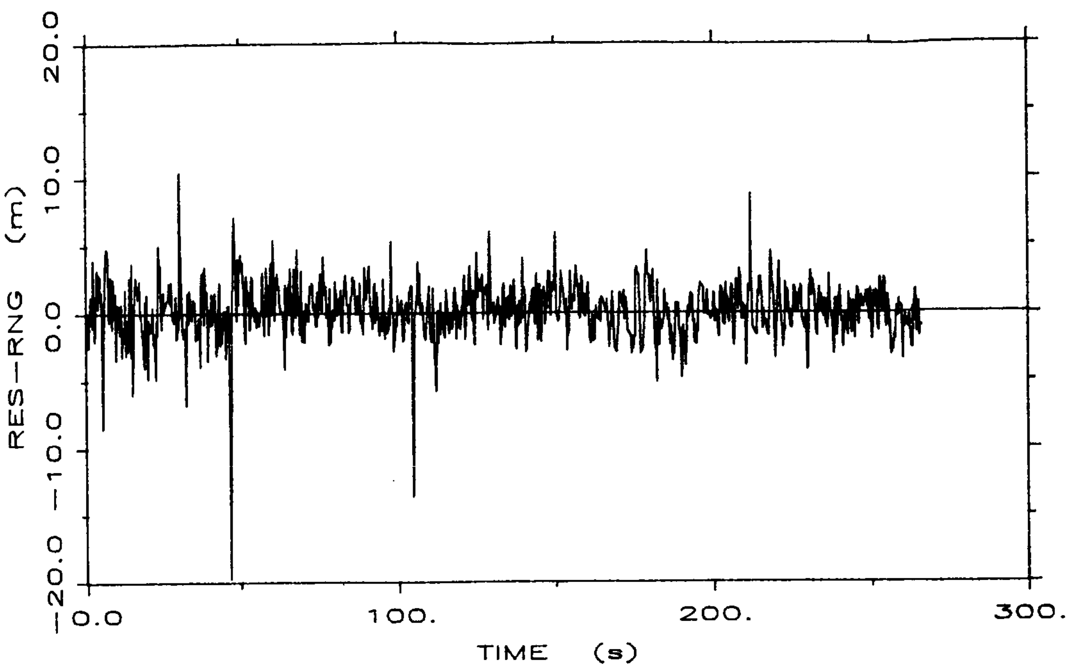


Figure 2.4: No-fail filter residuals for MLS azimuth and elevation



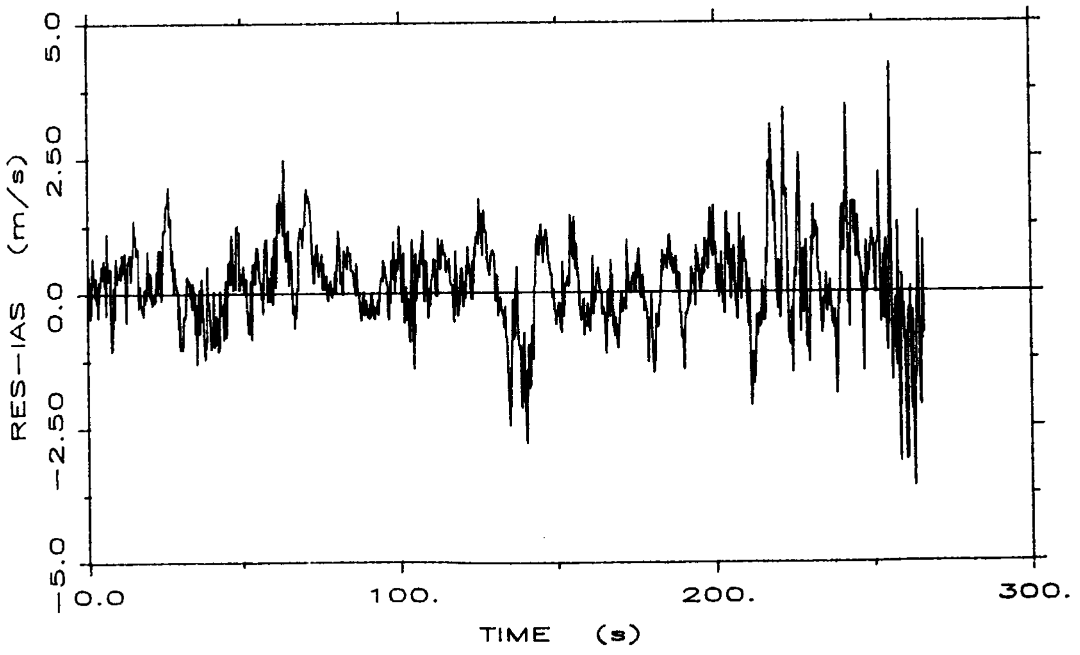


Figure 2.5: No-fail filter residuals for MLS range and IAS

while the wind estimates and accelerometer bias estimates are in the transient phase. The aircraft position and velocity estimates (not presented) are almost identical to earlier results.

Table 2.1 presents the empirical statistics for the no-fail filter residual sequences, calculated over the entire 266 seconds flight data run. These residual sequences still exhibit low sample means and standard deviations and thus continue to verify the good estimation performance of the NFF with the flight recorded data used.

Table 2.1: No-fail filter (NFF) residuals statistics: nominal 20 Hz run with auto-initialization routine

SENSOR	MEAN	S.D.	MAX	MIN	UNITS
MLS-Azim.	1.65E-03	7.41E-03	3.05E-02	-2.63E-02	deg
MLS-Elev.	4.56E-04	8.30E-03	3.29E-02	-2.48E-02	deg
MLS-Range	1.78E-01	2.02E+00	1.05E+01	-1.98E+01	m
IAS	1.60E-01	8.30E-01	4.66E+00	-4.19E+00	m/s
IMU-Roll	-1.46E-03	3.70E-02	1.29E-01	-1.32E-01	deg
IMU-Pitch	3.00E-03	2.47E-02	1.39E-01	-7.30E-02	deg
IMU-Yaw	4.14E-04	1.11E-01	6.33E-01	-4.54E-01	deg

2.2 New Isolation Logic and FDI Performance

As discussed in section 3.3 in [1], we had replaced the multiple hypothesis test in the FINDS algorithm performing simultaneous detection and isolation with a hierarchical detection and isolation test. The new set of mean detection tests over various moving window lengths of the averaged no-fail filter residuals was implemented to take advantage of the following:

(i) a bank of first order detectors equal to the total number of replicated active sensor channels would not have to be executed at every time instant, thus resulting in significant execution time savings;

(ii) the empirical mean of the averaged NFF residual sequences are always lower than those for the expanded NFF residuals, thus allowing a Chi-squared test of mean on these residuals;

(iii) on the basis of the incremental information behavior, various moving windows of these averaged NFF residuals could be tested, thus making the detection tests tuned to failures in various sensor subsets.

In [1], test results were presented summarizing the failure detection performance of this new strategy with failures injected into various sensors at three different times during the flight. However, the failure isolation and failure level estimation were not discussed as the isolation algorithm had not been updated to take into account the new detection strategy. Here, we present the subsequent changes made to the FINDS FDI algorithm and a series of results to validate these changes.

The multiple hypothesis test allowing simultaneous failure detection and isolation in the old strategy required a bank of first order Kalman filters, each hypothesizing a failure occurrence in a given replication of a given sensor. In the new strategy, these first order filters are activated only after the detection of a failure in order to isolate failures and estimate failure levels. This necessitated the following modifications to the old FDI module:

(i) Since the first order estimators are driven by the expanded innovations of the NFF, these expanded residuals need to be saved across the same moving windows as the averaged NFF residuals in the detection module.

(ii) Referring to equation 2.34a, 2.40, and 2.42 in [4], the bank of first order estimators require the covariance of the expanded residuals. This covariance is calculated using the prediction error covariance for the NFF states, the NFF gain and the expanded nonlinear measurement matrix at each time instant within a given detection window. To avoid the associated computational burden, these matrices are assumed to be constant across the entire detection window length in the new strategy. Our tests indicate that the incremental information to the various first order estimators does not degrade significantly under this assumption.

(iii) When the detector for moving window of length N samples declares a sensor failure, the isolation strategy assumes that the failure occurred exactly N samples prior to the detection. So the series of first order estimators are executed iteratively across the last N samples of the saved NFF expanded residuals using the current composite NFF matrices in assumption (ii) above. The failure compensated residuals generated by these filters drive the likelihood ratio computers iteratively and thus, at the end of N iterations we have a bank of likelihood ratios; the minimum of these ratios indicates which sensor has failed.

(iv) The incremental information analysis of [1], has shown that measurement sensor failures are reflected instantaneously in the NFF residuals whereas input sensor failures take longer to propagate through the NFF dynamics. Based on this incremental information behavior, we have selected three separate detector windows of lengths one, five, and ten samples. The knowledge of the different failure propagation speeds has been incorporated into the isolation module via the following. Since only a NFF measurement sensor failure can cause a failure declaration by the detector of window length one, only those first order estimators corresponding to NFF measurement sensors are executed under such a failure declaration. On the other hand, if the detector of length ten samples detects a failure, either one of the NFF input sensors or the IAS measurement sensor is assumed to have failed and the corresponding estimators are activated in the isolation module since any NFF measurement sensor failure would get detected before reaching the detector of window length ten. The window of five samples is capable of detecting either small bias failures in NFF measurement sensors or large bias failures in NFF input sensors; in such a case, all the first order filters are executed iteratively.

(v) After a failure is detected and isolated, the detector modules are not executed for N samples after an isolation. This prevents the use of corrupted residuals in the detection test until the appropriate window of NFF expanded residuals is filled again.

We now present the bias failure detection and isolation performance results for the composite FINDS algorithm covering the same set of runs described in Table 3.4a in [1] which covers only the detection performance without any isolation. The only difference is that with the auto-initialization routine implemented, the first iteration of flight data is now treated as t=0 seconds and used to initialize the NFF. Thus, the failure injection times get shifted by one sample or 0.05 seconds to ensure that the failures continue to be injected at the exact same instant in the actual flight.

Table 2.2a shows the results from the first set of runs which include a sensor failure occurring at 82.05 seconds into the flight -- note that all the bias estimates have converged by this time and the bank maneuver has not yet been executed. It is interesting to note that all of the input sensor failures except for the pitch rate gyro failure get detected by the detector of window length ten samples, while all the measurement sensor failures except the IAS sensor failure get detected instantaneously; i.e., by the detector of window length one sample. In general, the failure level estimation accuracy is high whenever the detection time is equal to the length of the moving detection window. Hence, the accuracy of the failure level estimate is high for the measurement sensors and moderate for the rate gyros. The large failure level estimation errors for the accelerometers are caused by the long detection times for these instruments which violate the assumption of the failure occurring ten samples prior to the detection. However, these inaccurate accelerometer failure level estimates do not have an unfavorable impact on the FINDS algorithm since failure level estimates are used only to increment the NFF covariance after the isolation of the failed sensor.

Table 2.2b, shows the results for the second series of runs in which single failures are injected into the flight data at 145.35 seconds during the runway alignment maneuver. Again, we see that the performance of the different window length detectors is as predicted with the larger windows allowing input sensor failure information to propagate into the residuals. In this set of runs, there are two interesting observations. In the case of the lateral accelerometer failure, a false isolation of the roll rate gyro occurred at t=150.10 seconds with a failure level estimate of 0.255 deg/s. This indicates that the incremental information available to the isolation strategy within the detection time span is insufficient for correct isolation. According to the implemented isolation logic, the detector of window length ten samples is inactivated up to t=150.60 seconds at which time it detects the propagating injected failure -- this time the isolation module does a correct isolation of the lateral accelerometer failure.

In the case of the IMU sensor failures, detection and correct isolation was followed by a second failure when the detector of window length ten samples was activated. Further investigation into the cause of this behavior points towards the NFF expanded residuals of the IMU pitch attitude sensors. When both replications of the IMU are active, the various detectors use the NFF residuals of the averaged attitude measurements, which are almost zero mean (see figures 2.11-12 in [1]. But, the individual expanded residuals of each IMU attitude sensor are biased due to normal dynamic errors during the aircraft bank maneuver. Thus when an IMU sensor failure is isolated, the reconfiguration logic throws out the entire IMU replication; the subsequent use of either of these biased residuals in the test-of-mean causes the detectors to indicate a failure. Thus, the reconfiguration logic for the IMU sensors includes compensation in the detectors for this biased behavior, and we do not see any occurrence of IMU false alarms now.

Table 2.2c consists of the third set of runs where the failures are injected during the aircraft final descent, or 238.65 seconds into the flight. The isolation strategy continues to show acceptable accuracy in terms of correct isolation of the failed sensor and also in terms of the estimation of the failure level. As in the second set of runs, the failed lateral accelerometer run resulted in a false isolation of the longitudinal accelerometer followed by a correct isolation. This is because the incremental information for the lateral accelerometer is the lowest of all sensors, referring back to figures 3.2a-d in [1].

In the second set of runs of Table 2.2b, the case of the roll rate gyro false alarm under a lateral accelerometer failure illustrates the need for

Table 2.2a: FDI performance with new isolation strategy: failures injected at 82.05 s

SENSOR TYPE	FAILURE DETECTED AT	[a]	DETECTION TIME	FAILURE LEVEL TRUE	ESTIM.	UNITS
Acc.-Long.	86.10 s	(10)	4.05 s	1.47	11.137	m/s/s
Acc.-Lat.	87.25 s	(10)	5.20 s	1.28	10.987	m/s/s
Acc.-Vert.	87.00 s	(10)	4.95 s	1.47	9.533	m/s/s
Gyro-Roll	82.50 s	(10)	0.45 s	0.90	0.831	deg/s
Gyro-Pitch	82.50 s	(5)	0.45 s	1.00	2.252	deg/s
Gyro-Yaw	83.65 s	(10)	1.60 s	1.00	2.401	deg/s
MLS-Azim.	82.05 s	(1)	0.0 s	0.18	0.173	deg
MLS-Elev.	82.05 s	(1)	0.0 s	0.18	0.181	deg
MLS-Range	82.05 s	(1)	0.0 s	40.00	40.482	m
IAS	82.50 s	(10)	0.45 s	9.00	10.239	m/s
IMU-Roll	82.05 s	(1)	0.0 s	1.50	1.412	deg
IMU-Pitch	82.05 s	(1)	0.0 s	2.00	1.862	deg
IMU-Yaw	82.05 s	(1)	0.0 s	4.00	4.087	deg

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

Table 2.2b: FDI performance with new isolation strategy: failures injected at 145.35 s

SENSOR TYPE	FAILURE DETECTED AT	[a]	DETECTION TIME	FAILURE LEVEL TRUE	FAILURE LEVEL ESTIM.	UNITS
Acc.-Long.	149.30 s	(10)	3.95 s	1.47	11.033	m/s/s
Acc.-Lat.	150.60 s	(10) [b]	5.25 s	1.28	10.544	m/s/s
Acc.-Vert.	149.20 s	(10)	3.85 s	1.47	10.256	m/s/s
Gyro-Roll	145.70 s	(5)	0.35 s	0.90	1.865	deg/s
Gyro-Pitch	145.85 s	(10)	0.50 s	1.00	1.051	deg/s
Gyro-Yaw	146.75 s	(10)	1.40 s	1.00	2.230	deg/s
MLS-Azim.	145.35 s	(1)	0.0 s	0.18	0.161	deg
MLS-Elev.	145.35 s	(1)	0.0 s	0.18	0.189	deg
MLS-Range	145.35 s	(1)	0.0 s	40.00	41.525	m
IAS	145.75 s	(10)	0.40 s	9.00	9.594	m/s
IMU-Roll	145.35 s	(1)	0.0 s	1.50	1.596	deg
IMU-Pitch	145.35 s	(1)	0.0 s	2.00	1.660	deg
IMU-Yaw	145.35 s	(1)	0.0 s	4.00	4.102	deg

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

[b] Detector of window=10 detected a failure at 150.10 s but it was falsely isolated as a gyro-roll failure of magnitude 0.255 deg/s. (Correct isolation of the injected failure occured at 150.60 s). The false alarm was healed at 153.60 s.

Table 2.2c: FDI performance with new isolation strategy: failures injected at 238.65 s

SENSOR TYPE	FAILURE DETECTED AT [a]		DETECTION TIME	FAILURE LEVEL TRUE	ESTIM.	UNITS
Acc.-Long.	242.40 s	(10)	3.75 s	1.47	10.109	m/s/s
Acc.-Lat.	241.85 s	(10) [b]	3.20 s	1.28	19.344	m/s/s
Acc.-Vert.	240.75 s	(10)	2.10 s	1.47	16.087	m/s/s
Gyro-Roll	239.05 s	(10)	0.40 s	0.90	0.811	deg/s
Gyro-Pitch	239.15 s	(10)	0.50 s	1.00	1.119	deg/s
Gyro-Yaw	240.15 s	(10)	1.50 s	1.00	2.505	deg/s
MLS-Azim.	238.65 s	(1)	0.0 s	0.18	0.185	deg
MLS-Elev.	238.65 s	(1)	0.0 s	0.18	0.179	deg
MLS-Range	238.65 s	(1)	0.0 s	40.00	40.905	m
IAS	239.85 s	(10)	1.20 s	9.00	9.751	m/s
IMU-Roll	238.65 s	(1)	0.0 s	1.50	1.478	deg
IMU-Pitch	238.65 s	(1)	0.0 s	2.00	2.109	deg
IMU-Yaw	238.65 s	(1)	0.0 s	4.00	4.084	deg

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

[b] Detector of window=10 detected a failure at 241.35 s but it was falsely isolated as a acc.-long. failure of magnitude -6.104 m/s/s. (Correct isolation of the injected failure occured at 241.85 s). The false alarm was healed at 244.85 s.

decoupling the translational dynamics related sensors from the rotational kinematics related sensors in the hierarchical FDI test. Even though the translational dynamics have been decoupled from the rotational kinematics in the no-fail filter, the isolation module is not able to take advantage of this fact. Only the NFF IMU residuals should govern the FDI of the rate gyros and attitude sensors, and the NFF MLS and IAS residuals should govern the FDI of the sensors concerned with the aircraft translational motion, i.e., accelerometers, MLS and IAS. This reasoning and the need to further improve the execution time has led us to partition the FINDS algorithm which is discussed in detail in the next chapter.

III. PARTITIONED FINDS AND MULTI-RATE IMPLEMENTATION

In this chapter, the first section deals with the partitioning of the FINDS algorithm into two sub-algorithms: one processing the sensors coupled with the translational dynamics and the other processing those sensors coupled with the rotational kinematics. Since the translational portion of the NFF had already been decoupled from the rotational portion [1], this split-up has no impact on the estimation performance of FINDS. Here, we present the test results showing the resultant reduction in execution time by the implementation of these two sub-algorithms in parallel on the dual-processor target flight computer. We also present a brief summary of the FDI performance for both of these FINDS subsets. In the second section, we discuss the changes made to the split-up version of FINDS in order to bring it to a real-time operation i.e., a multi-rate implementation in which the gains and covariance of the bias-free and bias filters in the NFF are updated at different frequencies. A table of execution times for various combinations of update frequencies is presented, followed by the FDI performance of one of the frequency combinations.

3.1 Partitioned FINDS

The composite FINDS algorithm with the combined translational dynamics and rotational kinematics has been discussed in [1] and in the previous chapter, with respect to its estimation performance, execution speed, program size and FDI performance. The composite NFF is of order 17 which includes 11 states in the bias-free filter (aircraft position, velocity, attitude and horizontal winds in the inertial runway frame) and six normal operating sensor biases estimated in the bias filter (linear accelerometer and rate gyro biases). This NFF uses seven measurements (MLS azimuth, elevation, range, IAS, IMU roll, pitch, yaw) thus necessitating the use of seventh order matrix operations in updating and propagating the filter estimates and also in the detection strategy. As reported in [1], at the nominal update frequency of 20 Hz., this composite FINDS algorithm with a program size of 116 Kb executed at a speed of approximately 11 times slower than real-time on one processor of the dual-processor configured target flight computer at a gain update frequency of 20 Hz. The execution speed was further increased to about 3 times slower than real-time by using a 4 Hz update frequency in the NFF--with some degradation in the estimation and FDI performance. Further modifications were obviously necessary to realize real-time execution.

One of the changes made in the NFF design in [1] was the use of a constant state transition matrix as opposed to a time-varying state dependent one [4], [5], which had to be updated by the partials of the input transition matrix at every iteration. The change not only reduced the execution time by 20% but also resulted in decoupling the NFF translational dynamics from the rotational kinematics. This fact along with the decision to retain the dual processor configured target flight computer in the current phase of the project has prompted us to partition the composite FINDS algorithm into two subsets to allow a real-time parallel processing solution as follows.

One computer (which we henceforth refer to as side 1) processes the sensors related to the the rotational kinematics: the rate gyros as NFF input sensors and the IMU attitude outputs as the NFF measurement sensors. The aircraft roll, pitch and yaw attitudes are the filter states in the bias-free portion of the NFF while the bias filter estimates the normal operating biases of the roll, pitch and yaw rate gyros. This 3-state and 3-bias configuration uses only three measurements viz. IMU roll, pitch and yaw; hence, only third order matrix operations are needed. On side 1, the suite of active sensors consists of dual replicated IMU sensors and one replication of the rate gyros yielding a total of nine sensors--the second replication of the rate gyros is kept in stand-by for activation in the event of a rate gyro failure. Thus, the failure detection test uses the NFF averaged residuals of the IMU attitude sensors across three moving windows of lengths one, five, ten samples while the new isolation strategy uses the NFF expanded IMU residuals in the isolation and failure level estimation process.

The second computer, referred to as side 2 henceforth in the report, processes the sensors related to the translational dynamics: the linear accelerometers as NFF input sensors, and MLS and IAS as NFF measurement sensors. The aircraft position, velocity and horizontal winds in the runway coordinates are the bias-free filter states while the bias filter estimates the accelerometer operating biases, thus resulting in an 8-state and 3-bias configuration. Since the MLS azimuth, elevation, range and IAS yield a total of four measurements, only fourth order matrix operations are performed on side 2 (with the exception of the eighth order state estimation covariance calculations). There are eight active sensors on this side comprised of single replications of the accelerometers and MLS and dual replication of the IAS sensor--again, the second replication of the accelerometer and MLS are kept in stand-by status.

The advantages of using such a partitioned parallel implementation are as follows:

(i) Since both modules are executed in parallel on the dual processor configuration of the target flight computer, this effectively involves only fourth order matrix operations at every sample as opposed to seventh order operations in the composite version--yielding a significant reduction in execution time.

(ii) On each side, the detection test is carried out for the sensors used by the NFF on that side only; the isolation test, thus, has to contend with only the active sensors in that particular module.

This effectively avoids false alarms across dynamically non-related sensors as in Table 2.2b in the previous chapter.

The interactions needed between the two individual subsets on each side are as follows:

(i) The rotational kinematics (side 1) need the NFF state estimates of the aircraft position and velocity from the translational dynamics side. These are used to calculate the current aircraft latitude/longitude and the coriolis and centripetal correction terms needed to compensate the platform gravity force (refer section 2.2.1 in [5]).

(ii) To augment the input vector, side 2 needs the gravity vector generated on side 1--this is calculated while estimating the aircraft current latitude and longitude and the correction terms of (i).

(iii) Side 2 also needs the NFF state estimates of the aircraft attitudes from side 1 to update the input transition matrix.

After porting the two subsets of the FINDS algorithm on to the two sides of the target flight computer, the interactions between the two sides have been achieved by using a DMA local data communication link. Also, side 1 was set-up to receive the incoming flight recorded data at every iteration, this DMA link was used to transfer the sensor data to side 2 as well. The data transfer rate of the DMA link is extremely high (2 Mb/s); hence, these interactions do not affect the overall execution time of either module to any noticeable extent.

Table 3.1a presents a brief overview of the difference between the composite and partitioned versions of FINDS in terms of program size. The reduction in program size from the composite to the split versions is mainly due to a reduction in the filter dimensions. Note also that the two subsets in the

Table 3.1a: Size characteristics: composite vs. partitioned FINDS

	Composite	Side 1	Side2
Program size	116 Kb	74 Kb	86 Kb
# of states	11	3	8
# of biases	6	3	3
# of inputs	6	3	3
# of measurements	7	3	4
# of active sensors	17	9	8
# of stand-by sensors	9	3	6

Table 3.1b: Speed characteristics: composite vs. partitioned FINDS

NFF Update Frequency	HOST DEVELOPMENT COMPUTER			TARGET FLIGHT COMPUTER Parallel Execution	
	Comp.	Side 1	Side2	with tape input	est. with DATAC
	(x real-time)				
20 Hz	9.00	1.30	3.55	3.55	3.20
10 Hz	5.37	0.92	2.05	2.18	1.80
5 Hz	3.44	0.75	1.37	1.47	1.10
4 Hz	3.05	0.72	1.23	1.32	0.95
2 Hz	1.77	0.66	0.91	1.05	0.67
1 Hz	1.40	0.63	0.78	0.91	0.53

NOTE : 1) On the target flight computer, all test runs were executed by using a magnetic tape read-in as the flight data interface. The estimated speed with the DATAC bus is obtained by discounting approximately 20 ms/cycle used by the tape drive.

2) All entries in the table are ratios of the total execution time on the indicated computer to the total simulated flight time of approximately 266 s.

dual configuration do not share any portion of the program code or matrix library subroutines--these are duplicated as needed on both sides.

The execution speed characteristics of the composite and partitioned FINDS algorithm shown in Table 3.1b exhibit the reduction in execution time by using the parallel implementation. Note that the entries in the table for the host development computer do not reflect a true parallel execution but individual execution times of each side. As for the target flight computer, the parallel execution speed is limited by the speed of the translational dynamics filter of side 2. Another important limiting factor is that for these test runs, a magnetic tape interface was used to read in the flight data at every iteration. This read-in interface is slow in comparison with the DATAC bus interface to be used in the actual flight test. Assuming a 20 ms/cycle reading time for the tape interface, the last column of this table reflects the estimated speed using the DATAC bus interface and a front-end microprocessor to perform variable assignments and the sensor dropout tests. Referring to the failure detection performance at various NFF update frequencies reported in [1], the lowest update frequency which yielded acceptable detection results was 4 Hz. In Table 3.1b, we see that at this frequency, the parallel implementation on the target flight computer is estimated to execute at faster than real-time speed.

Since the split-up has no impact of the NFF estimation performance of either side, we now present the FDI performance of the partitioned implementation for the same three sets of failure injection times given in the previous chapter for the composite algorithm. Tables 3.2a,b,c depict the performance of the partitioned FINDS algorithm in the parallel implementation with sensor failures injected at 82.05s, 145.35s and 238.65s into the same flight recorded data. Note that for this set of runs, the nominal update frequency of 20 Hz was chosen to allow a direct comparison with the results of Tables 2.2a,b, and

Table 3.2a: FDI performance of partitioned FINDS with nominal 20 Hz update frequency: failures injected at 82.05 s

SENSOR TYPE	FAILURE DETECTED AT	[a]	DETECTION TIME	FAILURE LEVEL TRUE	FAILURE LEVEL ESTIM.	UNITS
			ROTATIONAL KINEMATICS SIDE			
Gyro-Roll	82.45 s	(10)	0.40 s	0.90	0.727	deg/s
Gyro-Pitch	82.45 s	(5)	0.40 s	1.00	2.060	deg/s
Gyro-Yaw	83.45 s	(10)	1.40 s	1.00	2.175	deg/s
IMU-Roll	82.05 s	(1)	0.0 s	1.50	1.412	deg
IMU-Pitch	82.05 s	(1)	0.0 s	2.00	1.862	deg
IMU-Yaw	82.05 s	(1)	0.0 s	4.00	4.087	deg
			TRANSLATIONAL DYNAMICS SIDE			
Acc.-Long.	86.15 s	(10)	4.10 s	1.47	11.117	m/s/s
Acc.-Lat.	87.25 s	(10)	5.20 s	1.28	10.991	m/s/s
Acc.-Vert.	87.05 s	(10)	5.00 s	1.47	9.697	m/s/s
MLS-Azim.	82.05 s	(1)	0.0 s	0.18	0.173	deg
MLS-Elev.	82.05 s	(1)	0.0 s	0.18	0.181	deg
MLS-Range	82.05 s	(1)	0.0 s	40.00	40.483	m
IAS	82.50 s	(10)	0.45 s	9.00	10.239	m/s

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

Table 3.2b: FDI performance of partitioned FINDS with nominal 20 Hz update frequency: failures injected at 145.35 s

SENSOR TYPE	FAILURE DETECTED AT	[a]	DETECTION TIME	FAILURE LEVEL TRUE	ESTIM.	UNITS
			ROTATIONAL KINEMATICS SIDE			
Gyro-Roll	145.65 s	(5)	0.30 s	0.90	1.630	deg/s
Gyro-Pitch	145.80 s	(10)	0.45 s	1.00	0.927	deg/s
Gyro-Yaw	146.60 s	(10)	1.25 s	1.00	2.109	deg/s
IMU-Roll	145.35 s	(1)	0.0 s	1.50	1.596	deg
IMU-Pitch	145.35 s	(1)	0.0 s	2.00	1.660	deg
IMU-Yaw	145.35 s	(1)	0.0 s	4.00	4.102	deg
			TRANSLATIONAL DYNAMICS SIDE			
Acc.-Long.	149.35 s	(10)	4.00 s	1.47	11.267	m/s/s
Acc.-Lat.	150.30 s	(10)	4.95 s	1.28	11.022	m/s/s
Acc.-Vert.	149.25 s	(10)	3.90 s	1.47	11.196	m/s/s
MLS-Azim.	145.35 s	(1)	0.0 s	0.18	0.161	deg
MLS-Elev.	145.35 s	(1)	0.0 s	0.18	0.189	deg
MLS-Range	145.35 s	(1)	0.0 s	40.00	41.525	m
IAS	145.75 s	(10)	0.40 s	9.00	9.594	m/s

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

Table 3.2c: FDI performance of partitioned FINDS with nominal 20 Hz update frequency: failures injected at 238.65 s

SSENSOR TYPE	FAILURE DETECTED AT [a]		DETECTION TIME	FAILURE LEVEL TRUE	ESTIM.	UNITS
			ROTATIONAL KINEMATICS SIDE			
Gyro-Roll	239.00 s	(10)	0.35 s	0.90	0.717	deg/s
Gyro-Pitch	239.05 s	(5)	0.40 s	1.00	1.958	deg/s
Gyro-Yaw	239.95 s	(10)	1.30 s	1.00	2.256	deg/s
IMU-Roll	238.65 s	(1)	0.0 s	1.50	1.478	deg
IMU-Pitch	238.65 s	(1)	0.0 s	2.00	2.109	deg
IMU-Yaw	238.65 s	(1)	0.0 s	4.00	4.084	deg
			TRANSLATIONAL DYNAMICS SIDE			
Acc.-Long.	242.75 s	(10)	4.10 s	1.47	11.394	m/s/s
Acc.-Lat.	241.90 s	(10) [b]	3.25 s	1.28	22.079	m/s/s
Acc.-Vert.	240.80 s	(10)	2.15 s	1.47	17.295	m/s/s
MLS-Azim.	238.65 s	(1)	0.0 s	0.18	0.185	deg
MLS-Elev.	238.65 s	(1)	0.0 s	0.18	0.179	deg
MLS-Range	238.65 s	(1)	0.0 s	40.00	40.905	m
IAS	239.85 s	(10)	1.20 s	9.00	9.751	m/s

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

[b] Detector of window=10 detected a failure at 241.35 s which was falsely isolated as an IAS failure of magnitude 3.115 m/s. (Correct isolation of the injected failure occured at 241.90 s). The false alarm was healed at 244.90 s.

c in the previous chapter. On the rotational kinematics side, we see that there is a general reduction in the failure detection time (with an appropriate improvement in the failure level estimation) in almost every case. This is caused by the lower thresholds in the Chi-squared test-of-mean detection strategy due to a lower degree-of-freedom and also the low mean in the IMU NFF residuals. On the other hand, the translational dynamics side FDI performance remains essentially the same as previous results. Here, even though we have a lower degree-of-freedom, the detection thresholds cannot be reduced proportionately because of the unusually large residuals in MLS range and IAS--hence, the similar results.

Tables 3.3a,b, and c present the FDI performance of the partitioned parallel FINDS implementation for the same three sets of runs but using the NFF update frequency of 4 Hz, i.e., updating the bias-free and bias filter gains and covariances at a 4 Hz rate. At this low update rate, the NFF estimation performance (especially the bias filter performance) is not as good as the 20 Hz update case [1], yet the detection test and isolation strategy continue to portray acceptable performance. On side 1, the detection times and failure level estimates are comparable to those for the 20 Hz update runs presented in tables 3.2a, b, and c. The IMU failures continue to get detected instantaneously by the detector of window length one sample, resulting in extremely accurate failure level estimates. The rate gyro failures take slightly longer to get detected in the initial stages of the flight because of the higher uncertainty in the bias filter at the 4 Hz update frequency.

On side 2, Table 3.3c indicates a false isolation of the longitudinal accelerometer for the lateral accelerometer injected failure--this behavior can be attributed to the low incremental information available to the isolation strategy and also to the transients in the bias filter uncertainty at the 4 Hz update frequency. The MLS sensor failures during each of the three

Table 3.3a: FDI performance of partitioned FINDS with 4 Hz update frequency : failures injected at 82.05 s

SENSOR TYPE	FAILURE DETECTED AT [a]		DETECTION TIME	FAILURE LEVEL TRUE	ESTIM.	UNITS
			ROTATIONAL KINEMATICS SIDE			
Gyro-Roll	82.45 s	(10)	0.40 s	0.90	0.737	deg/s
Gyro-Pitch	82.45 s	(5)	0.40 s	1.00	2.060	deg/s
Gyro-Yaw	83.55 s	(10)	1.50 s	1.00	2.245	deg/s
IMU-Roll	82.05 s	(1)	0.0 s	1.50	1.412	deg
IMU-Pitch	82.05 s	(1)	0.0 s	2.00	1.860	deg
IMU-Yaw	82.05 s	(1)	0.0 s	4.00	4.091	deg
			TRANSLATIONAL DYNAMICS SIDE			
Acc.-Long.	88.35 s	(10)	5.30 s	1.47	10.838	m/s/s
Acc.-Lat.	- - - - - -	n.d.	- - - - -	1.28	- - -	m/s/s
Acc.-Vert.	93.25 s	(10)	11.20 s	1.47	7.333	m/s/s
MLS-Azim.	82.05 s	(1)	0.0 s	0.18	0.182	deg
MLS-Elev.	82.05 s	(1)	0.0 s	0.18	0.182	deg
MLS-Range	82.05 s	(1)	0.0 s	40.00	40.213	m
IAS	82.50 s	(10)	0.45 s	9.00	10.147	m/s

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

Table 3.3b: FDI performance of partitioned FINDS with 4 Hz update frequency: failures injected at 145.35 s

SENSOR TYPE	FAILURE DETECTED AT	[a]	DETECTION TIME	FAILURE LEVEL TRUE	ESTIM.	UNITS
			ROTATIONAL KINEMATICS SIDE			
Gyro-Roll	145.70 s	(10)	0.35 s	0.90	0.784	deg/s
Gyro-Pitch	145.80 s	(10)	0.45 s	1.00	0.964	deg/s
Gyro-Yaw	146.60 s	(10)	1.25 s	1.00	2.119	deg/s
IMU-Roll	145.35 s	(1)	0.0 s	1.50	1.603	deg
IMU-Pitch	145.35 s	(1)	0.0 s	2.00	1.649	deg
IMU-Yaw	145.35 s	(1)	0.0 s	4.00	4.094	deg
			TRANSLATIONAL DYNAMICS SIDE			
Acc.-Long.	150.20 s	(10)	4.85 s	1.47	8.257	m/s/s
Acc.-Lat.	151.15 s	(10)	5.80 s	1.28	12.699	m/s/s
Acc.-Vert.	149.95 s	(10)	4.60 s	1.47	14.011	m/s/s
MLS-Azim.	145.35 s	(1)	0.0 s	0.18	0.161	deg
MLS-Elev.	145.35 s	(1)	0.0 s	0.18	0.183	deg
MLS-Range	145.35 s	(1)	0.0 s	40.00	41.457	m
IAS	- - - - - -	n.d.	- - - - -	9.00	- - -	m/s

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

Table 3.3c: FDI performance of partitioned FINDS with 4 Hz update frequency: failures injected at 238.65 s

SENSOR TYPE	FAILURE DETECTED AT	[a]	DETECTION TIME	FAILURE LEVEL TRUE	FAILURE LEVEL ESTIM.	UNITS
			ROTATIONAL KINEMATICS SIDE			
Gyro-Roll	239.00 s	(10)	0.35 s	0.90	0.725	deg/s
Gyro-Pitch	239.05 s	(5)	0.40 s	1.00	1.965	deg/s
Gyro-Yaw	239.95 s	(10)	1.30 s	1.00	2.245	deg/s
IMU-Roll	238.65 s	(1)	0.0 s	1.50	1.477	deg
IMU-Pitch	238.65 s	(1)	0.0 s	2.00	2.108	deg
IMU-Yaw	238.65 s	(1)	0.0 s	4.00	4.088	deg
			TRANSLATIONAL DYNAMICS SIDE			
Acc.-Long.	243.75 s	(10)	5.10 s	1.47	7.514	m/s/s
Acc.-Lat.	242.05 s	(10) [b]	3.40 s	1.28	20.415	m/s/s
Acc.-Vert.	240.95 s	(10)	2.30 s	1.47	16.853	m/s/s
MLS-Azim.	238.65 s	(1)	0.0 s	0.18	0.184	deg
MLS-Elev.	238.65 s	(1)	0.0 s	0.18	0.179	deg
MLS-Range	238.65 s	(1)	0.0 s	40.00	40.912	m
IAS	239.90 s	(10)	1.25 s	9.00	9.775	m/s

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

[b] Detector of window=10 detected a failure at 241.55 s which was falsely isolated as a acc-long. failure of magnitude -8.44 m/s/s. (Correct isolation of the injected failure occured at 242.05 s). The false alarm was healed at 245.05 s.

flight segments are detected and isolated with the same accuracy and detection speed as in the nominal runs. Except for the maneuver segment, the IAS sensor failures are also detected and isolated accurately. During the maneuver, the errors due to nonlinearities (caused by the low NFF update frequency) and the dynamic wind model absorb part of the injected bias in the IAS sensor. This results in that particular failure being not detected; a subsequent test run with a higher injected failure level (15 m/s) resulted in a FDI performance equivalent to the run with 20 Hz update frequency. It is also interesting to see the performance of the healer algorithm in all the cases involving either false detection or false isolation; in each case, the sensor in question is healed at the end of the healing window length of 3 seconds.

3.2 FINDS Multi-Rate Implementation

In chapter 2 in [1], we had presented the estimation performance of the FINDS NFF using various update frequencies between 20 Hz and 1 Hz. Satisfactory state estimation was noted even for the low update frequency of 1 Hz by monitoring the statistics of the NFF residual sequences. However, the bias filter performance degraded as the update frequency was lowered due to the increased time taken by the bias estimates to converge to steady-state values.

The FDI performance of FINDS with the lower NFF update frequencies, as given in chapter 3 in [1] and in the previous section, reveals that the input sensor FDI is affected the most; the detection times are large in most cases and the injected failure goes undetected in some instances of the accelerometer failures. An analysis shows that this behavior is caused by the longer delay times of the bias estimation uncertainty; a high prediction error covariance at the time of the injected failure results in the bias estimates tracking the added bias failure and converging to new steady-state values. This

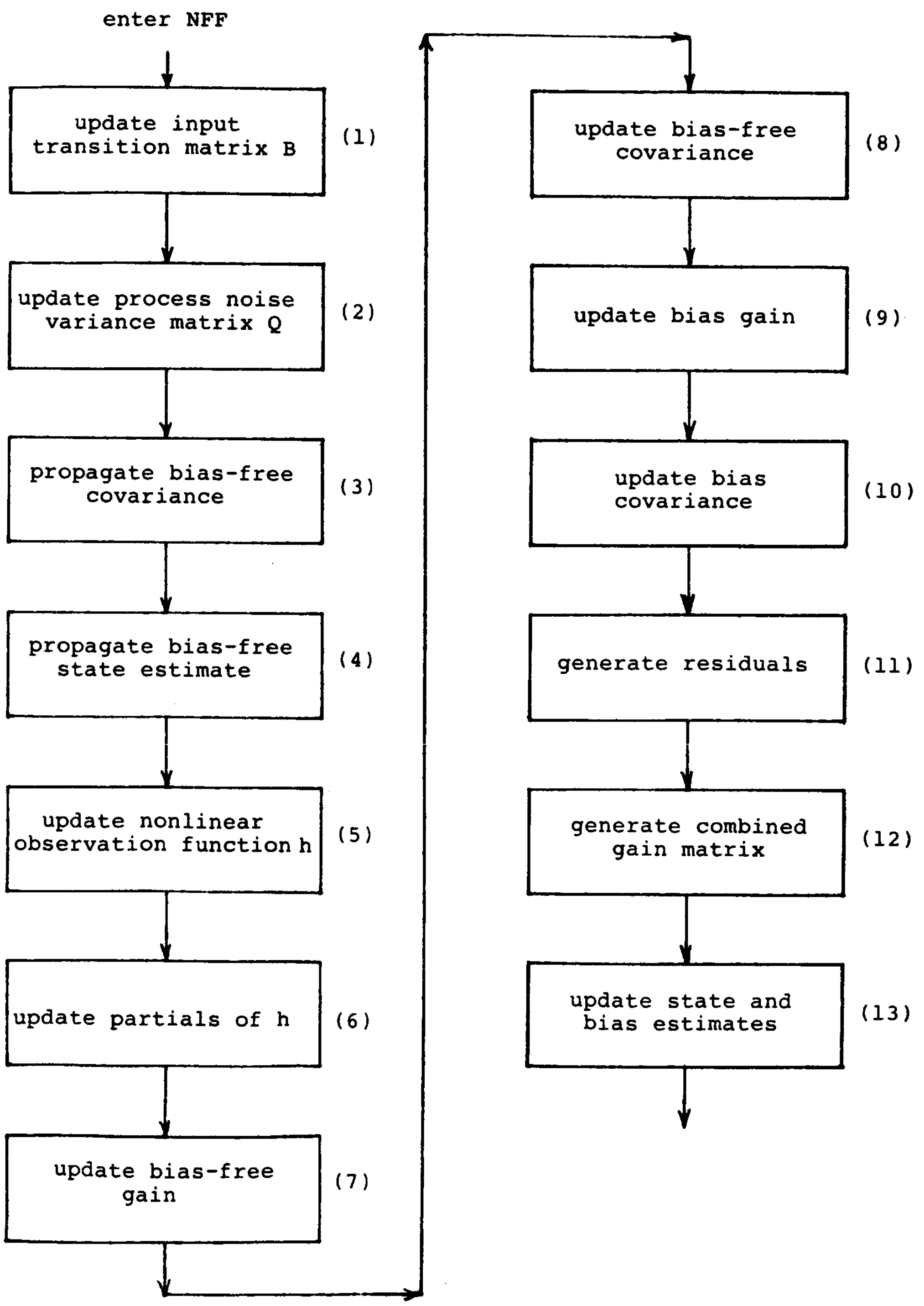

enter NFF
update input transition matrix B
(1)
update process noise variance matrix Q
(2)
propagate bias-free covariance
(3)
propagate bias-free state estimate
(4)
update nonlinear observation function h
(5)
update partials of h
(6)
update bias-free gain
(7)
update bias-free covariance
(8)
update bias gain
(9)
update bias covariance
(10)
generate residuals
(11)
generate combined gain matrix
(12)
update state and bias estimates
(13)

Figure 3.1: FINDS NFF flow chart

fact has led us to study the effect of using higher update frequencies for the bias filter covariance than for the other portions of the NFF.

Figure 3.1 shows the chronological order of the various tasks to be performed in the NFF via a flow chart representation; by executing all the blocks numbered 1 through 13 in the estimation process, the nominal 20 Hz update frequency is realized. We can improve the bias estimation performance by updating the bias filter covariance at a higher frequency than the remainder of the NFF in one of two ways:

(A) Bias-free vs. bias filter split

In this multi-rate structure, the bias filter is executed at a higher frequency than the bias-free filter. Referring to figure 3.1, this implies executing blocks 2, 3, 6, 7, 8 at one frequency and blocks 9, 10 at higher update rates. Note that blocks 1, 4, 5, 11, 12, 13 need to be executed at every iteration regardless of the update frequencies in use.

(B) Gain vs. covariance update split

In this multi-rate structure, the bias-free and bias filter gains are updated at one frequency while the respective covariances are updated at a higher frequency. Thus, blocks 6, 7, 9 are updated at a lower frequency than blocks 2, 3, 8, 10; again, blocks 1, 4, 5, 11, 12, 13 have to be executed at a 20 Hz update rate.

Table 3.4 gives a summary of the execution speed achieved by the composite FINDS algorithm and the partitioned FINDS algorithm for various frequency combinations in options (A) and (B) above. In the case of the bias-free/bias filter frequency split, the choice of update rates is governed by these conditions:

(i) the bias filter update frequency should be greater than the bias-free filter update rate,

Table 3.4: Execution speed summary: bias-free/bias and gain covariance split

Frequency Split	HOST DEVELOPMENT COMPUTER Comp.	Side 1	Side2	TARGET FLIGHT COMPUTER Parallel Execution with tape input	est. with DATAC
	(x real-time)				
BIAS-FREE/BIAS SPLIT					
5 Hz / 20 Hz	5.55	1.02	1.99	1.77	1.40
1 Hz / 20 Hz	4.72	0.97	1.66	1.66	1.28
5 Hz / 1 Hz	4.36	0.86	1.62	1.62	1.24
1 Hz / 10 Hz	3.55	0.80	1.29	1.16	0.79
1 Hz / 5 Hz	2.64	0.70	1.03	1.02	0.64
GAIN/COVARIANCE SPLIT					
5 Hz / 20 Hz	6.62	0.98	2.72	3.05	2.67
1 Hz / 20 Hz	6.09	0.92	2.60	2.99	2.62
5 Hz / 10 Hz	4.70	0.83	1.85	2.05	1.67
1 Hz / 10 Hz	4.15	0.76	1.70	1.85	1.47
1 Hz / 5 Hz	2.90	0.68	1.21	1.36	0.99

NOTE : 1) All entries in the table are ratios of the total execution time on the indicated computer to the total simulated flight time of approximately 266 s.

(ii) the bias-free filter should be updated at 5 Hz or lower for real-time execution.

The first set of these runs shows that we have an estimated real-time execution on the target flight computer when the bias-free filter runs at a 1 Hz frequency and the bias filter is updated at 10 Hz or less.

The second part of the table deals with the gain/covariance frequency split. Here again, the choice of update rate is governed by,

(i) both bias-free and bias covariances should be updated at faster rates than the respective gains,

(ii) the gain update rates should be 5 Hz or lower for real-time execution.

This second set of runs reveals an estimated real-time run on the target flight computer for only the 1 Hz/5 Hz gain/covariance split. Another interesting point in both sets of runs is that the rotational kinematics side (side 1) executes faster than real-time in all the cases except the 5 Hz/20 Hz bias-free/bias split. The translational dynamics side is the limiting factor in the real-time set-up; hence, side 1 can be configured with almost any frequency combination without losing real-time performance. This is important because both the estimation performance and FDI performance can be optimized for real-time operation with the appropriate choice of update frequencies.

The execution speed results presented so far in this report for the partitioned FINDS algorithm show that side 2 executes about two times slower than side 1 and hence is the limiting factor in a real-time realization. Run-time analysis performed on the various blocks of Figure 3.1 indicate that the maximum amount of time in a given cycle is spent in block 8 of side 2; i.e., in updating the bias-free covariance of the translational dynamics filter. This is primarily due to the number of states involved (8 states) which results in eighth order matrix manipulations. In order to run this block at a different frequency, we need a multi-rate implementation where the four quantities viz bias-free filter gain (gnx), bias-free filter covariance (cvx), bias filter gain (gnb), and bias filter covariance (cvb) are all updated at different rates.

Using the FDI results of Tables 3.3a-c and combining the constraints of options A and B above, we have the following boundary conditions on the update rates: (denoting f as frequency)

(i) $f_{cvb} \geq f_{cvx}$

(ii) $f_{gnb} \geq f_{gnx}$

(iii) $f_{cvb}, f_{cvx} \geq 4$ Hz

(iv) $f_{gnb}, f_{gnx} \leq 4$ Hz

(v) discrete $f_{cvx}, f_{cvb}, f_{gnx}, f_{gnb} \in$ [1Hz,2Hz,4Hz,5Hz,10Hz,20Hz]

The five constraints above yield about eighteen different combinations of update rates for the four portions of the NFF on either side. Table 3.5 presents the execution speed summary for these possible combinations. It is seen that at least six of these choices result in a real-time end-to-end execution on side 2; side 1 continues to execute at better than real-time speed for all the frequency permutations in the table.

We now consider a sample parallel implementation using different multi-rate update frequencies on both sides to ensure an end-to-end real-time run. The NFF estimation performance of both sides is discussed and we also present the FDI performance of both subsets for the three sets of failure runs as before.

Tables 3.1b, 3.4, and 3.5 indicate that the rotational kinematics side executes faster than real-time for all the frequency combinations except the nominal 20 Hz update case. Thus, we can choose a 10 Hz update rate for f_{cvx},

Table 3.5: Execution speed summary: multi-rate updates

Update Frequencies Fcvx,Fcvb,Fgnx,Fgnb (Hz)	HOST DEVELOPMENT COMPUTER Comp.	Side 1	Side2	TARGET FLIGHT COMPUTER Parallel Execution with tape input	est. with DATAC
	(x real-time)				
10 , 20 , 2 , 4	4.83	0.89	1.98	2.03	1.66
5 , 20 , 2 , 4	4.07	0.84	1.64	1.55	1.17
4 , 20 , 2 , 4	3.91	0.84	1.57	1.41	1.04
5 , 10 , 2 , 4	3.14	0.76	1.37	1.42	1.05
4 , 10 , 2 , 4	2.99	0.76	1.31	1.30	0.92
4 , 4 , 2 , 4	2.53	0.72	1.17	1.25	0.87
10 , 20 , 1 , 4	4.78	0.88	1.97	2.02	1.65
5 , 20 , 1 , 4	4.01	0.84	1.62	1.53	1.15
4 , 20 , 1 , 4	3.86	0.83	1.55	1.40	1.03
5 , 10 , 1 , 4	3.09	0.76	1.35	1.41	1.03
4 , 10 , 1 , 4	2.94	0.76	1.29	1.30	0.92
4 , 5 , 1 , 4	2.48	0.71	1.15	1.24	0.86
10 , 20 , 1 , 2	4.62	0.86	1.93	2.00	1.62
5 , 20 , 1 , 2	3.85	0.83	1.60	1.50	1.12
4 , 20 , 1 , 2	3.69	0.81	1.53	1.39	1.02
5 , 10 , 1 , 2	2.92	0.74	1.32	1.39	1.02
4 , 10 , 1 , 2	2.77	0.73	1.25	1.27	0.89
4 , 5 , 1 , 2	2.31	0.69	1.11	1.20	0.83

NOTE : 1) All entries in the table are ratios of the total execution time on the indicated computer to the total simulated flight time of approximately 266 s.

f_{cvb}, f_{gnx}, and f_{gnb} on side 1 assuring the least degradation in both estimation and FDI performance from the nominal run. This choice yields an execution speed of 0.92 times real-time (refer Table 3.1b) on the host development computer and an estimated similar or faster speed on the target flight computer with the use of the DATAC data bus interface. On side 2, we see from Table 3.5 that a choice of six multi-frequency combinations are possible to ensure a real-time end-to-end execution. For our test run implementation, we have selected the 4 Hz, 10 Hz, 2 Hz, and 4 Hz combination for the update rates for f_{cvx}, f_{cvb}, f_{gnx}, and f_{gnb}, respectively. This

Table 3.6: NFF residuals statistics for sample parallel implementation with multi-rate updates

Side 1 Update Rates: Fcvx=10 Hz, Fcvb=10 Hz, Fgnx=10 Hz, Fgnb=10 Hz

Side 2 Update Rates: Fcvx= 4 Hz, Fcvb=10 Hz, Fgnx= 2 Hz, Fgnb= 4 Hz

SENSOR	MEAN	S.D.	MAX	MIN	UNITS
		ROTATIONAL KINEMATICS SIDE			
IMU-Roll	-7.36E-04	3.69E-02	1.29E-01	-1.30E-01	deg
IMU-Pitch	1.47E-03	2.46E-02	1.36E-01	-7.47E-02	deg
IMU-Yaw	-5.28E-04	1.11E-01	6.28E-01	-4.57E-01	deg
		TRANSLATIONAL DYNAMICS SIDE			
MLS-Azim.	2.91E-03	7.49E-03	3.42E-02	-2.74E-02	de
MLS-Elev.	1.42E-04	7.28E-03	3.15E-02	-2.21E-02	deg
MLS-Range	1.74E-01	1.97E+00	1.03E+01	-1.94E+01	m
IAS	1.18E-01	7.66E-01	4.65E+00	-4.19E+00	m/s

choice yields an estimated speed of 0.92 times real-time for side 2 on the target flight computer in conjunction with the faster DATAC interface.

Table 3.6 presents the NFF residuals statistics as a measure of estimation performance of the two sides, in the sample parallel implementation outlined above. In comparison with the nominal frequency composite algorithm estimation performance shown in Table 2.1, we see that the statistical behavior of these NFF residuals is very similar in the sense that they portray the same low means and standard deviations, thus indicating good NFF estimates. Time history plots of the bias/state estimates (not shown here) verified this assertion. The failure detection and isolation performance of this multi-rate parallel implementation is presented in the next three sets of results.

Table 3.7a presents the results of the first series of runs with the failure occurrence at 82.05 s. Keeping the 4 Hz update FDI performance of Table 3.3a as the acceptable baseline for a real-time run, the improvement here in the case of the accelerometer failures is immediately evident. The longitudinal and vertical accelerometers show a 10% and 25% faster detection time, respectively; the lateral accelerometer failure which went undetected in the 4 Hz case is now detected and correctly isolated 7.15 seconds after its injection. The MLS sensor failures on the translational dynamics side and the IMU sensor failure on the rotational kinematics side are detected and isolated instantaneously as in the nominal case. The IAS and rate gyro sensor FDI is also similar to that of the 20 Hz update run (refer Table 3.2a).

In the second set of runs, the results with sensor failures injected at 145.35 s are shown in Table 3.7b. Again, the accelerometer FDI performance shows faster detection by 0.1 to 0.25 seconds than for the 4 Hz update case of Table 3.3b. The MLS, IMU, and rate gyro sensor FDI continues to show the same excellent performance as the 20 Hz case of Table 3.2b. The IAS sensor failure

Table 3.7a: FDI performance for sample parallel implementation with multi-rate updates: failures injected at 82.05 s

Side 1 Update Rates: Fcvx=10 Hz, Fcvb=10 Hz, Fgnx=10 Hz, Fgnb=10 Hz

Side 2 Update Rates: Fcvx= 4 Hz, Fcvb=10 Hz, Fgnx= 2 Hz, Fgnb= 4 Hz

SENSOR TYPE	FAILURE DETECTED AT [a]	DETECTION TIME	FAILURE LEVEL TRUE	FAILURE LEVEL ESTIM.	UNITS
ROTATIONAL KINEMATICS SIDE					
Gyro-Roll	82.45 s (10)	0.40 s	0.90	0.734	deg/s
Gyro-Pitch	82.45 s (5)	0.40 s	1.00	2.064	deg/s
Gyro-Yaw	83.45 s (10)	1.40 s	1.00	2.163	deg/s
IMU-Roll	82.05 s (1)	0.0 s	1.50	1.411	deg
IMU-Pitch	82.05 s (1)	0.0 s	2.00	1.861	deg
IMU-Yaw	82.05 s (1)	0.0 s	4.00	4.088	deg
TRANSLATIONAL DYNAMICS SIDE					
Acc.-Long.	86.85 s (10)	4.85 s	1.47	11.194	m/s/s
Acc.-Lat.	89.20 s (10)	7.15 s	1.28	11.031	m/s/s
Acc.-Vert.	90.45 s (10)	8.40 s	1.47	11.423	m/s/s
MLS-Azim.	82.05 s (1)	0.0 s	0.18	0.183	deg
MLS-Elev.	82.05 s (1)	0.0 s	0.18	0.182	deg
MLS-Range	82.05 s (1)	0.0 s	40.00	40.256	m
IAS	82.50 s (10)	0.45 s	9.00	10.047	m/s

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

Table 3.7b: FDI performance for sample parallel implementation with multi-rate updates: failures injected at 145.35 s

Side 1 Update Rates: Fcvx=10 Hz, Fcvb=10 Hz, Fgnx=10 Hz, Fgnb=10 Hz

Side 2 Update Rates: Fcvx= 4 Hz, Fcvb=10 Hz, Fgnx= 2 Hz, Fgnb= 4 Hz

SENSOR TYPE	FAILURE DETECTED AT	[a]	DETECTION TIME	FAILURE LEVEL TRUE	ESTIM.	UNITS
			ROTATIONAL KINEMATICS SIDE			
Gyro-Roll	145.70 s	(10)	0.35 s	0.90	0.798	deg/s
Gyro-iitch	145.80 s	(10)	0.45 s	1.00	0.939	deg/s
Gyro-Yaw	146.60 s	(10)	1.25 s	1.00	2.111	deg/s
IMU-Roll	145.35 s	(1)	0.0 s	1.50	1.598	deg
IMU-Pitch	145.35 s	(1)	0.0 s	2.00	1.656	deg
IMU-Yaw	145.35 s	(1)	0.0 s	4.00	4.099	deg
			TRANSLATIONAL DYNAMICS SIDE			
Acc.-Long.	150.10 s	(10)	4.75 s	1.47	8.558	m/s/s
Acc.-Lat.	151.05 s	(10)	5.70 s	1.28	11.974	m/s/s
Acc.-Vert.	149.70 s	(10)	4.35 s	1.47	13.987	m/s/s
MLS-Azim.	145.35 s	(1)	0.0 s	0.18	0.164	deg
MLS-Elev.	145.35 s	(1)	0.0 s	0.18	0.184	deg
MLS-Range	145.35 s	(1)	0.0 s	40.00	41.850	m
IAS	- - - - - - - -	n.d.	- - - - - -	9.00	- - -	m/s

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

Table 3.7c: FDI performance for sample parallel implementation with multi-rate updates: failures injected at 238.65 s

Side 1 Update Rates: Fcvx=10 Hz, Fcvb=10 Hz, Fgnx=10 Hz, Fgnb=10 Hz

Side 2 Update Rates: Fcvx= 4 Hz, Fcvb=10 Hz, Fgnx= 2 Hz, Fgnb= 4 Hz

SENSOR TYPE	FAILURE DETECTED AT [a]		DETECTION TIME	FAILURE LEVEL TRUE	ESTIM.	UNITS
			ROTATIONAL KINEMATICS SIDE			
Gyro-Roll	239.00 s	(10)	0.35 s	0.90	0.715	deg/s
Gyro-Pitch	239.05 s	(5)	0.40 s	1.00	1.964	deg/s
Gyro-Yaw	239.95 s	(10)	1.30 s	1.00	2.255	deg/s
IMU-Roll	238.65 s	(1)	0.0 s	1.50	1.479	deg
IMU-Pitch	238.65 s	(1)	0.0 s	2.00	2.108	deg
IMU-Yaw	238.65 s	(1)	0.0 s	4.00	4.085	deg
			TRANSLATIONAL DYNAMICS SIDE			
Acc.-Long.	243.70 s	(10)	5.05 s	1.47	7.631	m/s/s
Acc.-Lat.	242.00 s	(10) [b]	3.35 s	1.28	20.422	m/s/s
Acc.-Vert.	240.95 s	(10)	2.30 s	1.47	16.964	m/s/s
MLS-Azim.	238.65 s	(1)	0.0 s	0.18	0.184	deg
MLS-Elev.	238.65 s	(1)	0.0 s	0.18	0.179	deg
MLS-Range	238.65 s	(1)	0.0 s	40.00	40.891	m
IAS	239.90 s	(10)	1.25 s	9.00	9.779	m/s

[a] Numbers in parentheses indicate the length of the moving window for the detector which detected the failure.

[b] Detector of window=10 detected a failure at 241.50 s which was falsely isolated as an IAS failure of magnitude 3.455 m/s. (Correct isolation of the injected failure occured at 242.00 s). The false alarm was healed at 245.00 s.

is not detected here just as in the 4 Hz case because of the low failure level injected, nonlinear errors associated with the multi-rate updates, and the dynamic wind model.

Finally, Table 3.7c shows the results with sensor failures injected at 238.65 s, in the final runway approach segment of the flight emulation. Comparing with the 20 Hz update rate results of Table 3.2c, we see that except for the accelerometer failures, there is no degradation in FDI performance in the other sensors. The accelerometer failure detection times, however, continue to be a few samples lower than in the 4 Hz run of Table 3.3c.

Summarizing, in this chapter we have presented the implementation and performance of a partitioned version of FINDS on the target flight computer to take advantage of its dual-processor configuration. The NFF estimation performance is identical to the composite FINDS algorithm, the FDI performance is slightly better, and the execution time for the nominal 20 Hz run is reduced by a factor of three over the composite version using one processor. We have also presented implementations using multiple frequency updates for the gains and covariances of the bias-free and bias portions of the NFF and the effect of various combinations on execution speed. Finally, we have taken a sample multi-rate parallel implementation which yields a total execution time lower than the simulated flight time on the target flight computer. The FDI performance of this implementation is shown to be much better than the performance for the single gain update frequency of 4 Hz.

IV. CONCLUSIONS AND RECOMMENDATIONS

In this report, we have presented the modifications made to the FINDS algorithm in order to match its isolation test with the new detection strategy reported in [1], to facilitate testing with different flight data sequences, to port the software onto a dual-processor configured target flight computer, and to increase its execution speed to allow real-time operation on the test-bed target computer.

The isolation algorithm has been modified to conform with the test-of-mean detection strategy described in the previous report [1]. Since the detectors are able to detect NFF input sensor failures quicker, the isolation strategy has to cope with reduced amounts of incremental information for performing a successful isolation at the time of detection. The isolation logic which still consists of the bank of first order Kalman filters has been modified to account for the length of the detector window flagging the failure. Results of this new isolation test with the same three series of failure injected flights have been presented, and indicate acceptable performance in isolating the faulty sensor and estimating its failure level.

An auto-initialization routine has been incorporated into the FINDS algorithm so that the NFF estimates can be appropriately initialized for different flight test sequences--this routine uses the first iteration of the flight data to generate the NFF initial state conditions.

For the dual-processor configured target flight computer chosen as the candidate test-bed computer, we have partitioned the composite FINDS algorithm into two parts: one processing the sensors related to the rotational kinematics and the other processing the sensors related to translational dynamics. These sub-algorithms have been successfully ported onto the two sides of the target flight computer. With the use of a DMA data link to transfer relevant program variables between the two sides, the parallel execution of the partitioned algorithm has been demonstrated. By partitioning the FINDS algorithm, its execution speed has increased from about 11 times slower than real-time to 3.5 times slower than real-time for the nominal 20 Hz update run.

Necessary software changes have been incorporated in the FINDS algorithm to realize a multi-rate implementation, i.e., to perform the bias-free and bias filter gain and covariance computations at different frequencies and thus to obtain an ideal compromise between real-time execution speed and good estimation and FDI performance. The results with a sample choice of update frequencies on both sides of the parallel implementation have been presented. This multi-rate implementation yields better than real-time end-to-end execution speed with an acceptable FDI performance for all sensor failures except for accelerometers.

Based on the results obtained, we recommend the following:

1) An end-to-end real-time execution capability of the partitioned FINDS algorithm has been demonstrated on the target flight computer with the use of multi-frequency updates. We now recommend performing run-time analysis to obtain individual cycle timing diagrams at the various frequencies. In particular, real-time rate tree structures need to be developed to facilitate a time-phased implementation of the multi-rate updates and thus to ensure real-time execution for each cycle.

2) We recommend the evaluation of the partitioned multi-rate FINDS algorithm with the use of the nonlinear six degree-of-freedom NASA ATOPS B-737 simulation in an MLS environment. In view of the limited availability of actual flight recorded sensor data, this would facilitate the testing of the current FINDS algorithm with different flight paths and atmospheric conditions such as discrete gusts and shears. Since the "true" aircraft states would be available from the simulation, we

can test the FINDS NFF for estimation accuracy and address the issue of using these estimates in the aircraft flight control system.

3) Instead of the current target flight computer, we recommend the use of one of the various commercially available faster computers as a test-bed for flight test experiments. This would allow the use of the nominal 20 Hz frequency in all execution runs and thus obviate the small degradation in estimation and FDI performance which is inherent to the multi-rate update implementation.

4) The current isolation strategy with its bank of first order estimators utilizes more cycle time than a real-time implementation would allow. Further study of the isolation routine is needed in order to develop a new and simpler isolation test which would execute within the constraints of individual cycle times.

5) We recommend the sequence of flight test experiments detailed in chapter 4 in [1] to validate the performance of the current parallel implementation. In particular, these tests would constrain the isolation algorithm to dynamically related sensor groups, thus minimizing the computational burden without making the FDI problem any easier.

REFERENCES

[1] Caglayan, A.K. and Godiwala, P.M., "A Preliminary Design for Flight Testing the FINDS Algorithm," NASA CR-178043, March 1986.

[2] Caglayan, A.K. and Godiwala, P.M., "Evaluation of a Fault Tolerant System for an Integrated Avionics Sensor Configuration with TSRV Flight Data," NASA CR-172589, June 1985.

[3] Pines, S., Schmidt, S.F. and Mann, F., "Automated Landing Rollout, and Turnoff using MLS and Magnetic Cable Sensors," NASA CR-2907, October 1977.

[4] Caglayan, A.K. and Lancraft, R.E., "An Aircraft Sensor Fault Tolerant System," NASA CR-165876, April 1982.

[5] Caglayan, A.K. and Lancraft, R.E., "A Fault Tolerant System for an Integrated Avionics Sensor Configuration," NASA CR-3834, September 1984.

[6] Caglayan, A.K., Godiwala, P.M. and Morrell, F.R., "Performance Analysis of a Fault Inferring Nonlinear Detection System (FINDS) with Integrated Avionics Flight Data," Proceedings of AIAA Computers in Aerospace V Conference, Long Beach, CA, October 1985.

Standard Bibliographic Page

1. Report No. NASA CR-178252	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle A DUAL-PROCESSOR MULTI-FREQUENCY IMPLEMENTATION OF THE FINDS ALGORITHM		5. Report Date April 1987
		6. Performing Organization Code
7. Author(s) Pankaj M. Godiwala and Alper K. Caglayan		8. Performing Organization Report No. R8610
9. Performing Organization Name and Address Charles River Analytics Inc. 55 Wheeler Street Cambridge, MA 02138		10. Work Unit No.
		11. Contract or Grant No. NAS1-17719
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, DC 20546		13. Type of Report and Period Covered Contractor Report
		14. Sponsoring Agency Code 505-66-41-05

15. Supplementary Notes
Langley Technical Monitor: Frederick R. Morrell
Second Supplemental Final Report

16. Abstract

This report presents a parallel processing implementation of the FINDS (Fault Inferring Nonlinear Detection System) algorithm on a dual processor configured target flight computer. First, a filter initialization scheme is presented which allows the no-fail filter (NFF) states to be initialized using the first iteration of the flight data. A modified failure isolation strategy, compatible with the new failure detection strategy reported earlier, is discussed and the performance of the new FDI algorithm is analyzed using flight recorded data from the NASA ATOPS B-737 aircraft in a Microwave Landing System (MLS) environment. The results show that low level MLS, IMU, and IAS sensor failures are detected and isolated instantaneously, while accelerometer and rate gyro failures continue to take comparatively longer to detect and isolate. The parallel implementation is accomplished by partitioning the FINDS algorithm into two parts: one based on the translational dynamics and the other based on the rotational kinematics. Finally, a multi-rate implementation of the algorithm is presented yielding significantly low execution times with acceptable estimation and FDI performance.

17. Key Words (Suggested by Authors(s)) Fault Tolerant Systems, Parallel processing, Sensor failure detection and isolation.		18. Distribution Statement Unclassified - Unlimited Subject Category 06	
19. Security Classif.(of this report) Unclassified	20. Security Classif.(of this page) Unclassified	21. No. of Pages 61	22. Price A05