# A Framework for Collaborative All-Source Navigation with Fault Detection and Exclusion

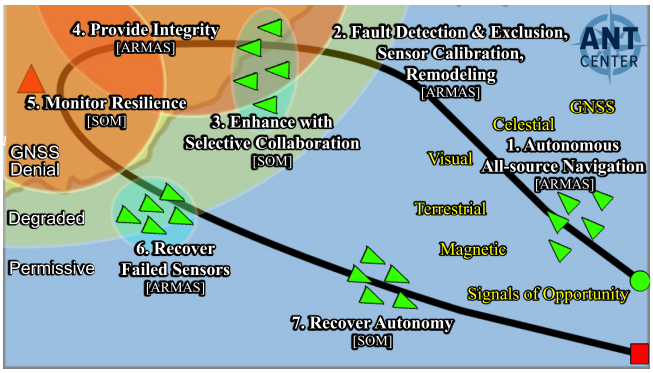


Fig. 1. The ARMAS-SOM framework provides intrinsic all-source resilience "awareness" via novel selective offboard collaboration which improves navigation integrity and facilitates graceful recovery to autonomy.

**Jonathon S. Gipson**, Student Member, IEEE
Air Force Test Pilot School, USA

**Robert C. Leishman**, Member, IEEE
Air Force Institute of Technology, USA

***Abstract*— The ARMAS-SOM framework fuses collaborative all-source sensor information in a resilient manner with fault detection, exclusion, and integrity solutions recognizable to a GNSS user. This framework uses a multi-filter residual monitoring approach for fault detection and exclusion which is augmented with an additional "observability" Extended Kalman Filter (EKF) sub-layer for resilience. We monitor the aposteriori state covariances in this sub-layer to provide intrinsic awareness when navigation state observability assumptions required for integrity are in danger. The framework leverages this to selectively augment with offboard information and preserve resilience. By maintaining split parallel *collaborative* and *proprioceptive* instances and employing the "stingy collaboration" technique, we are able maximize efficient use of network resources, limit the propagation of unknown corruption to a single donor, and maintain consistent collaborative navigation without fear of double-counting in a scalable processing footprint. Lastly, we preserve the ability to return to autonomy and are able to use the same intrinsic awareness to notify the user when it is safe to do so.**

***Index Terms*—collaborative, distributed, all-source, navigation, detection, integrity**

## I. Introduction

Due to widespread awareness of Global Navigation Satellite System (GNSS) vulnerabilities, there is increased interest in alternative navigation technologies such as Visual [1], Signals of Opportunity [2], Magnetic [3], and others [4][5][6]. The fusion of these information sources for localization is known as *all-source navigation* [7]. Since all-source sensors typically reside on individual vehicles connected in a distributed wireless network, consistent collaborative fusion of this decentralized information is highly desired.

A primary navigation user requirement is a timely and accurate estimate of navigation error to include impending degradation known as *navigation integrity* [8] which is currently provided through GNSS techniques such as Receiver Autonomous Integrity Monitoring (RAIM) [9] and Advanced Receiver Autonomous Integrity Monitoring (ARAIM). Current state of the art Fault Detection and Exclusion (FDE) techniques employed for GNSS fault detection and integrity monitoring include both solution separation and measurement residual (innovation) methods [10] [11]. These methods have been successfully employed with measurements from multiple GNSS constellations which resulted in worldwide ARAIM coverage without the need for additional ground infrastructure. [12]. With GNSS navigation, simultaneously redundant, synchronous measurements with validated measurement models are readily available. With all-source navigation, none of these are guaranteed. This research forms a framework to provide FDE and integrity monitoring using asynchronous, heterogeneous all-source navigation sensors in a distributed collaborative vehicle network.

The following section provides a brief history of all-source navigation to provide context for this research. We continue with brief overviews of classical model estimation, residual monitoring and applicability to multi-sensor FDE. The background section ends with a brief explanation of estimator credibility. These concepts are fundamental to the ARMAS framework [13] which is augmented by Stable Observability Monitoring (SOM) [14]. The next section describes methods for managing cross-correlation and introduces split *collaborative* and *proprioceptive* ARMAS-SOM modules which enable consistent collaborative augmentation with the option to gracefully recover autonomy. These improvements are simulated in a 3D all-source navigation environment which demonstrates collaborative estimator credibility, resilience to loss of state observability, and navigation error improvement.

14 Jul 2021

Jonathon S. Gipson was with the Autonomy and Navigation Technology Center, U.S. Air Force Institute of Technology, WPAFB, OH 43524, USA. He is now with the United States Air Force Test Pilot School, USA (e-mail: jonathon.gipson@us.af.mil). Dr. Robert C. Leishman is the Director of the Autonomy and Navigation Technology Center, U.S. Air Force Institute of Technology, WPAFB, OH 43524, USA (e-mail: robert.leishman@afit.edu).

## II. Background

### A. Recursive Bayesian Estimation

Small cumulative measurement errors form an integration bias or "drift", the primary error source for an Inertial Navigation System (INS). Some form of recursive model estimation, like an EKF, is often used to maintain an INS error model. This strategy incorporates trusted external GNSS and/or geodetic measurements to periodically correct integration bias. Navigation is based on accurate estimation of a vehicle's system states. This is accomplished via a task called model estimation.

Modern navigation, based on recursive model estimation, can trace its roots to the Kalman Filter (KF) algorithm [15]. In a Kalman filter, the system states, $\mathbf{x}$, are estimated recursively in real-time by propagating a state estimate, $\hat{\mathbf{x}}$, and associated state covariance, $\mathbf{P}$. The process (dynamics) model is given by

$$\dot{\mathbf{x}}(t) = \mathbf{F}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{G}\mathbf{w}(t), \tag{1}$$

with

$$E[\mathbf{w}(t)] = 0, \tag{2}$$

$$E[\mathbf{w}(t)\mathbf{w}(t)^T(t+\tau] = \mathbf{Q}\delta(\tau), \tag{3}$$

where $\mathbf{x}$ is the system state vector, $\mathbf{u}$ is the system input control vector, and $\mathbf{w}$ is a vector of white noise components. $\mathbf{F}$, $\mathbf{B}$, and $\mathbf{G}$ are linear operator matrices for the state vector, control input vector, and noise vector, respectively.

The Van Loan [16] provides a method to discretize 1 and 3 for use in digital, time-sampled computer systems. The resulting discrete process noise strength matrix, $\mathbf{Q}_d$, discrete control input matrix, $\mathbf{B}_d$, and discrete state transition matrix, $\mathbf{\Phi}$, are calculated by linearizing the system about a single time sample, $\Delta t$.

Assuming state estimate observability, measurements, $\mathbf{z}$, and associated measurement covariances, $\mathbf{R}$, are used to perform a state estimate update. The measurements are mapped to the states by the observation model, $\mathbf{H}$. The discrete measurement model is given by

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \tag{4}$$

where

$$E[\mathbf{v}_k] = 0, \tag{5}$$

$$E[\mathbf{v}_k\mathbf{v}_{k+\Delta t}^T] = \mathbf{R}\delta_{k,k+\Delta t}, \tag{6}$$

and $\mathbf{z}$ is the sensor measurement vector. The discrete Kalman filter is initialized with state estimates, $\mathbf{x}_0$, and associated initial state covariance $\mathbf{P}_0$. The initial state estimate of $\hat{\mathbf{x}}$ is propagated in the discrete Kalman filter using

$$\hat{\mathbf{x}}_{\mathbf{k+1}}^{-} = \mathbf{\Phi}\hat{\mathbf{x}}_{\mathbf{k}}^{+} + \mathbf{B_d}\mathbf{u_k}, \tag{7}$$

$$\mathbf{P}_{\mathbf{k+1}}^{-} = \mathbf{\Phi}\mathbf{P}_{\mathbf{k}}^{+}\mathbf{\Phi}^{\mathbf{T}} + \mathbf{Q_d}. \tag{8}$$

This update is performed by optimally combining the stochastic components of the state estimate and of the measurements via the Kalman gain, $K$, using

$$\mathbf{K_k} = \mathbf{P_k^-}\mathbf{H}^{\mathbf{T}}[\mathbf{H}\mathbf{P_k}\mathbf{H}^T + \mathbf{R}]^{-1}, \tag{9}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k[\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^-], \tag{10}$$

$$\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\mathbf{P}_k^-. \tag{11}$$

With properly modeled state-transition and measurement models, the result is optimal stochastic estimation of the vehicle states. The state-transition model makes use of mathematical relationships between system states to provide additional observability to states which cannot be directly measured. Recursive estimation is the primary means of model estimation used by this research to approach the all-source navigation problem.

### B. Residual Monitoring

This brief overview of residual monitoring based on a likelihood function for a Kalman filter or EKF is based on a more thorough introduction [17]. Residual monitoring is a useful technique for sensor fault detection. The goal of this section is to set the stage for a later explanation of the Sensor-Agnostic All-source Residual Monitoring (SAARM) algorithm implemented by the Autonomous and Resilient Management of All-source Sensors (ARMAS) framework. After a sensor measurement $\mathbf{z}$ is obtained, it is possible to extract a set of pre-update KF residuals, $\mathbf{r}$, at time $k$, resulting in

$$\mathbf{r}_k = \mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^-. \tag{12}$$

If we apply Gaussian white-noise assumptions, the expected statistical distribution of these residuals is

$$\mathbf{r}_k \hookrightarrow \mathcal{N}\left(\mathbf{0}, \mathbf{H}\mathbf{P}_k^-\mathbf{H}^{\mathrm{T}} + \mathbf{R}\right), \tag{13}$$

which results in a Gaussian-Normal probability density function given by

$$p(\mathbf{r}_k|\mathbf{\Sigma}_k) = \frac{1}{\sqrt{|2\pi\mathbf{\Sigma}_k|}} e^{-\frac{1}{2}(\mathbf{r}_k)^{\mathrm{T}}\mathbf{\Sigma}_k^{-1}(\mathbf{r}_k)} \text{ and} \tag{14}$$

$$\mathbf{\Sigma}_k = \mathbf{H}\mathbf{P}_k^-\mathbf{H}^{\mathrm{T}} + \mathbf{R}. \tag{15}$$

If we assume a set of $N$ residuals have been collected preceding $t_k$, the resulting likelihood function is given by

$$L_{N_k} = \prod_{j=k-N+1}^{k} \frac{1}{\sqrt{|2\pi\mathbf{\Sigma}_j|}} e^{-\frac{1}{2}(\mathbf{r}_j)^{\mathrm{T}}\mathbf{\Sigma}_j^{-1}(\mathbf{r}_j)}. \tag{16}$$

In lieu of differentiation, this expression can be simplified by taking advantage of the monotonically increasing natural logarithm function, resulting in the log-likelihood function given by

$$\mathcal{L}_{N_k} = \sum_{i=k-N+1}^{k} -\log(|2\pi\mathbf{\Sigma}_i|) - \frac{1}{2}\mathbf{r}_i^{\mathrm{T}}\mathbf{\Sigma}_i^{-1}\mathbf{r}_i. \tag{17}$$

If we examine a residual vector $\mathbf{r_k}$ containing the $N$ most recent residuals, the log-likelihood function acts as a moving window to compile the cumulative statistical likelihood for this set. A predefined threshold can be used to determine if the set of residuals falls outside the expected distribution which would trigger a failure detection. This forms the basis for the fault detection and exclusion algorithm espoused by ARMAS-SOM.

## III. Autonomous Resilient Management of All-source Sensors (ARMAS)

Introduced in 2018, ARMAS provides a generalized framework for real-time management of heterogeneous, asynchronous all-source sensors [13]. This framework is resilient to corruption from mismodeled, uncalibrated, and faulty sensors and is accomplished by combining sensor validation, FDE, recalibration, and remodeling modes in a single architecture. ARMAS employs a set of SCORPION pluggable EKF estimators [18] to address the following nonlinear navigation problem:

$$\dot{\mathbf{x}}(t) = \boldsymbol{f}[\mathbf{x}(t), \boldsymbol{\epsilon}(t), \boldsymbol{u}(t), t] + \boldsymbol{G}(t)\boldsymbol{w}(t) \tag{18}$$

where $\mathbf{x}$ is a $N \times 1$ state vector of a vehicle's position, velocity, and attitude. The measurement error states vector $\epsilon$ is of dimension $M \times 1$, $\mathbf{u}$ is the control input vector, $\mathbf{G}$ is an $(N+M) \times W$ linear operator, and $\mathbf{w}$ is a $W \times 1$ white noise process defined by a $W \times W$ continuous process noise strength matrix, $\mathbf{Q}$.

The state estimates are propagated through optimally combining the state process model, sensor-specific calibration parameters, and measurement updates from j = $1...J$ available all-source sensors. The measurement model for the $j^{th}$ sensor is described by:

$$\mathbf{z}_k^{[j]} = \mathbf{h}^{[j]}\left[\mathbf{x}(t), \epsilon^{[j]}(t), \mathbf{u}(t), t, \mathbf{p}^{[j]}\right] + \boldsymbol{v}_k^{[j]}, \tag{19}$$

where $\mathbf{h}^{[j]}$ is the nonlinear measurement function for the $j^{th}$ sensor, $\epsilon^{[\mathbf{j}]}$ is an $L \times 1$ subset of $\epsilon$ which contains additional error states needed to process sensor measurements, $\mathbf{p}^{[\mathbf{j}]}$ is a $P \times 1$ user-selectable model parameter vector for $\mathbf{h}^{[j]}$, and $\mathbf{v_k}$ is a $Z \times 1$ discrete white noise process with covariance defined by matrix $\mathbf{R}_{\mathbf{k}}^{[\mathbf{j}]}$.

The $Z \times 1$ measurement residual for sensor $j$, $\mathbf{r}_{\mathbf{k}}^{[\mathbf{j}]}$ is defined by

$$\mathbf{r}_k^{[j]} = \mathbf{z}_k^{[j]} - \mathbf{h}^{[j]}\left[\hat{\mathbf{x}}_k^-, \hat{\epsilon}_k^{[j]-}, \mathbf{u}_k, t_k, \hat{\mathbf{p}}_k^{[j]}\right], \tag{20}$$

where $\hat{\mathbf{x}}_k^-$, $\hat{\epsilon}_k^{[j]-}$, and $\hat{\mathbf{p}}_k^{[j]}$ are estimated quantities. Assuming white Gaussian noise, the measurement residual from 20 is expected to follow the distribution

$$\mathbf{r}_k^{[j]} \hookrightarrow \mathcal{N}\left(\mathbf{0}_{N\times 1}, \mathbf{S}_k^{[j]}\right), \tag{21}$$

$$\mathbf{S}_k^{[j]} = \mathbf{H}_k^{[j]}\mathbf{P}_k^-\mathbf{H}_k^{[j]^T} + \mathbf{R}_k^{[j]}, \tag{22}$$

where $\mathbf{P}_k^-$ is the $(N+M) \times (N+M)$ state estimate error covariance matrix at time $t_k$ and $\mathbf{H}_k^{[j]^T}$ is the $Z \times (N+M)$ Jacobian of $\mathbf{h}^{[j]}$.

Sensors are initialized in one of two modes: trusted or untrusted. Untrusted sensors are required to enter a sensor validation mode prior to being brought into monitoring mode. In validation mode, ARMAS employs a likelihood function described by (17) to monitor the statistical distribution of a user-defined monitoring period composed of recent Kalman pre-update residuals. A Chi-square, $\chi^*$, test statistic is used to detect excursions outside a user-defined threshold across the sampling period. Although validation mode and monitoring mode use the same detectors, sensors in validation mode are excluded from impacting the main state estimates. Once a a sensor passes validation mode, it becomes trusted. Trusted sensors are brought online into monitoring mode. In monitoring mode, sensor measurements are allowed to update the main state estimates. ARMAS employs the same pre-update residual likelihood function used in the validation mode to monitor sensor performance. A detailed explanation of monitoring mode, including FDE and integrity functions is given in section A.

Once a fault is detected, the sensor is no longer "trusted" and is quarantined from affecting the core navigation state estimate, $\hat{\mathbf{x}}^{[j]}$. ARMAS attempts to reinitialize the sensor via validation mode. If this fails, ARMAS attempts to repair and recover the faulty sensor via two separate modes: sensor calibration and remodeling.

In calibration mode, user-selectable sensor parameters, $\mathbf{p}^{[j]}$ and/or $\epsilon^{[j]}$ are estimated using residual monitoring from trusted sensors that have observability of $\mathbf{x}$. If there is a single calibration parameter, ARMAS attempts to correct the calibration using residual monitoring and sends the sensor back to validation mode. If linked extrinsic calibration parameters exist, (e.g. camera lever arm and camera orientation within $\mathbf{p}^{[j]}$ or $\epsilon^{[j]}$), these are estimated individually and sequenced based on convergence of the state covariance matrix to maintain state observability.

If the recalibrated sensor fails to pass sensor validation, the sensor enters remodeling mode where ARMAS attempts to modify the measurement model, $\mathbf{h}^{[j]}$, based on $1...S$ user-defined measurement models. $S$ concurrent filters are spawned (each with a unique measurement model), and an epoch of measurement residuals is gathered against the core navigation estimate $\mathbf{x}$. The 'winning' sensor measurement model is selected based on which filter best matches the prescribed distribution (21) during the residual epoch. The sensor then enters validation mode. If the remodeling mode does not result in a new model selection, and Resilient Sensor Recovery (RSR) is activated, the sensor periodically re-enters validation mode after a user-selectable time period in an attempt to overcome a temporal anomaly. Figure 2 is a state transition diagram depiction of these modes. The result is a framework compatible with heterogeneous, asynchronous all-source sensors with the benefits of resilience against various sensor calibration, modeling, and temporal faults.

ARMAS requires overlapping state observability [19] to detect anomalous sensor behavior. Sensor suites without overlapping observability for a particular estimated state cannot be used for fault detection and exclusion because the residuals are unrelated (e.g. barometric altimeter, accelerometer, and airspeed sensor). As discussed above, the system monitors Kalman pre-update residuals between sensor measurements and subfilter estimates to continuously judge whether sensor measurements adhered to the distribution prescribed by the sensor model. Anomalous sensor behaviors (e.g. bias, gain, model mismatch, high noise, etc.) are only observable if there are other sensors with comparable observability into the state estimate. If anomalous behavior is detected, the ARMAS

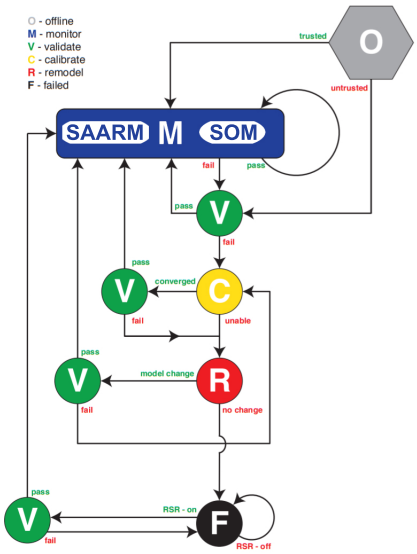


Fig. 2. ARMAS Framework State Diagram with SAARM and SOM identified [13]

framework attempts to recover the sensor through recalibration, remodeling, and re-validation. Without overlapping state observability, it is impossible to determine if a sensor is misbehaving or if it can be re-validated.

### A. Sensor-Agnostic All-source Residual Monitoring (SAARM)

As previously alluded, the monitoring mode in ARMAS uses a special form of residual monitoring. SAARM is designed to detect multiple sensor failure modes: bias, mismatched model, and/or miscalibration. For resilience to a single faulty sensor, this approach requires the designer to maintain $J = I$ differently configured navigation subfilters (one for each sensor). This is later extended to provide resiliency to multiple simultaneous faults (at the expense of processing power). The key to SAARM's FDE and integrity functions is the use of overlapping state observability to detect a faulty sensor. The pluggable Bayesian filters provided by the SCORPION [18] estimation architecture afford needed flexibility to spawn, propagate, and remove multiple concurrent filters on the fly. The following section summarizes SAARM's fault detection, fault exclusion, and all-source integrity methods.

SAARM assumes a system form of

$$\dot{\mathbf{x}}(t) = \boldsymbol{f}[\mathbf{x}(t), \boldsymbol{\epsilon}(t), \boldsymbol{u}(t), t] + \boldsymbol{G}(t)\boldsymbol{w}(t) \tag{23}$$

where $\mathbf{x}$ is a $N \times 1$ state vector of a vehicle's position, velocity, and attitude. The measurement error states vector $\epsilon$ is of dimension $M \times 1$, $\mathbf{u}$ is the control input vector, $\mathbf{G}$ is an $(N+M) \times W$ linear operator, and $\mathbf{w}$ is a $W \times 1$ white noise process defined by a $W \times W$ continuous process noise strength matrix, $\mathbf{Q}$. SAARM estimates system states with $J$ separate subfilters. At time $t = t_k$, the system state vector and state estimation covariance matrix are defined by

$\hat{\mathbf{x}}^{[j]}(t_k)$ and $\mathbf{P}^{[j]}_{\hat{\mathbf{x}}\hat{\mathbf{x}}}(t_k)$ for $j = 1...J$ separate subfilters.

Each of these subfilters is informed by a subset of $I-1$ sensors. At $t = t_k$, the $i^{th}$ sensor provides measurements given by

$$\mathbf{z}^{[i]}(t_k) = \mathbf{h}^{[i]}\left[\hat{\mathbf{x}}^{[j]}(t_k^-), \mathbf{u}(t_k), t_k\right] \tag{24}$$

where $\mathbf{h}^{[i]}$ is the nonlinear measurement function, $\mathbf{u}(t_k)$ is the control input function, and $\mathbf{v}^{[i]}(t_k)$ is a discrete white noise process of dimension $\mathbf{Z}_i \times 1$ defined by covariance matrix $\mathbf{R}^{[i]}(t_k)$. The pre-update measurement estimate for sensor $i$ from filter $j$ is defined by

$$\hat{\mathbf{z}}^{[i,j]}(t_k) = \mathbf{h}^{[i]}\left[\hat{\mathbf{x}}^{[j]}(t_k^-), \mathbf{u}(t_k), t_k\right], \tag{25}$$

where the estimated covariance matrix is defined by

$$\mathbf{P}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}{}^{[i,j]}(t_k^-) = \mathbf{H}^{[i]}(t_k^-)\mathbf{P}_{\hat{\mathbf{x}}\hat{\mathbf{x}}}{}^{[j]}(t_k^-)\mathbf{H}^{[i]}(t_k^-)^T. \tag{26}$$

Using (25) and (26), the "pre-update residual" vector between sensor $i$ and filter $j$, $\mathbf{r}^{[i,j]}$ and its covariance matrix, $\mathbf{P}^{[i,j]}_{rr}$ are defined as

$$\mathbf{r}^{[i,j]}(t_k) = \mathbf{z}^{[i]}(t_k) - \hat{\mathbf{z}}^{[i,j]}(t_k^-), \tag{27}$$

$$\mathbf{P}^{[i,j]}_{rr}(t_k) = \mathbf{R}^{[i]}(t_k) + \mathbf{P}^{[i,j]}_{\hat{z}\hat{z}}(t_k^-). \tag{28}$$

Fault detection relies on computing a moving average of recent residual-space test statistics formed by pre-update residual vectors from (27) and (28). ARMAS is designed to detect three categories of faults: (1) a bias, (2) an incorrectly stated noise covariance, or (3) an incorrectly stated measurement model. The likelihood function focuses on a single residual-space statistic derived from the Mahalanobis distance $d$ given by

$$d^2 = (\mathbf{y} - \mu)^T \boldsymbol{\Sigma}^{-1} (\mathbf{y} - \mu), \tag{29}$$

where $\mu$ and $\boldsymbol{\Sigma}$ are the mean and covariance of a $Z_i$-dimensional Gaussian distribution and $\mathbf{y}$ is the sensor measurement. It is known that a sum of $M$ independent $d^2$ distances follows a $\chi^*$ distribution with $Z$ degrees of freedom [20] given by

$$\chi^* = \sum_{s=k}^{k+M} d^2(t_s), \tag{30}$$

$$d^2(t_k) = \mathbf{r}^T(t_k)[\mathbf{P}_{rr}(t_k)]^{-1}\mathbf{r}(t_k). \tag{31}$$

The set of pre-update residuals is known to be a zero-mean, white sequence [17]. The fault detection test for $M$ pre-residuals is composed of the following hypotheses:

$$H_0 : \chi^*_{[i,j]} < \chi^2(1 - \alpha/2, M \times Z_i) \tag{32}$$

$$H_1 : \chi^*_{[i,j]} > \chi^2(1 - \alpha/2, M \times Z_i) \tag{33}$$

where $\alpha$ is the probablity of false alarm and $M$ is the number of averaged pre-residual samples. A Type 1 error of $\alpha$ = 0.05 is chosen for this research. $H_0$ refers to the hypothesis where the fault is not present in filter $j$. $H_1$ refers to the hypothesis where a fault is present in filter $j$. The resulting hypothesis test forms the basis of the fault detection algorithm.

Once a fault is detected, a consensus of multiple subfilters is utilized to exclude the faulty sensor. With $J = I$ subfilters, SAARM can only exclude single

faults within each residual monitoring epoch (i.e. $M$-sample moving average). Since this technique relies on a single measurement epoch, insidious errors with very slow growth may not be detected. In this scenario, each subfilter is informed by a different subset of $I-1$ sensors (i.e. each subfilter is missing a single sensor). SAARM also assumes that all states are observable by all subfilters. In addition to $J=I$ subfilters, a main filter is maintained to generate a full navigation state estimate for user output. Accordingly, cross-covariance terms between the main filter and any other filters are not used for any computation. For this scenario, SAARM provides an axiom for fault exclusion: *under the assumption that, at most, one sensor can fail simultaneously, at least one of the $J$ subfilters will be completely unaffected by faulty measurements* [13].

The fault consensus is tallied in a $\mathbf{T}$-matrix of dimension $I \times J$ and uses the following convention:

$$\mathbf{T}(i,j) = \begin{cases} 0, \text{ Sensor } i \text{ not associated with filter } j \\ 0, \ \chi^*_{[i,j]} < \chi^2(1-\alpha/2, M \times Z_i), \text{ No Fault, } H_0 \\ 1, \ \chi^*_{[i,j]} > \chi^2(1-\alpha/2, M \times Z_i), \text{ Fault, } H_1 \end{cases}$$

Figure 4 shows the relationship between $I$ sensors and $J$ subfilters required for "fault consensus" sensor exclusion. The rows correspond to the $i=1...I$ sensors and the columns correspond to the $j=1...J$ subfilters. Each row contains measurements, $\mathbf{Z}^{[i]}$, and the measurement error covariance matrix, $\mathbf{R}^{[i]}$, from the $i^{th}$ sensor. Each column contains the estimated measurement, $\hat{\mathbf{z}}^{[i,j]}$, and its error covariance matrix, $\mathbf{P}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}{}^{[i,j]}$.

Based on the stated convention, a fault is declared when $\mathbf{T}$ contains a single nonzero entry (i.e. at least one subfilter detected $H_1$). After a fault is declared, SAARM waits for a consensus from the remaining subfilters until a single fault-free subfilter remains. It is assumed that the last remaining fault-free subfilter is the one which does not contain the faulty sensor. After fault exclusion, the fault-free subfilter is elevated to "main filter" status and the pre-update residual monitoring epoch is restarted with $I-1$ total sensors and $J-1$ subfilters. Each newly spawned subfilter now contains $I-2$ sensors. The faulty sensor is removed from monitoring mode and follows the state diagram shown in Figure 2. Of note, SAARM is able to detect the occurrence of multiple simultaneous faults but is not able to provide simultaneous fault exclusion with the minimum quantity of $I=J$ subfilters.

For SAARM to properly handle multiple simultaneous faults, multiple layers of subfilters are required. The number of concurrent subfilters, $J_N$, required to handle $N$ simultaneous faults for $I$ sensors is:

$$J_N = \binom{I}{I-N} = \frac{I!}{N!(I-N)!} \tag{34}$$

As one might expect, the subfilter (and processing) requirements to guarantee resiliency to multiple faults is non-trivial. For example, an 8 sensor system resilient to 2 simultaneous faults (occuring within the same pre-residual monitoring epoch) requires 28 concurrent subfilters. Since this construct requires 8 layer one subfilters and 1 main filter, a total of 37 concurrent filters is required to support the 8 sensor system.

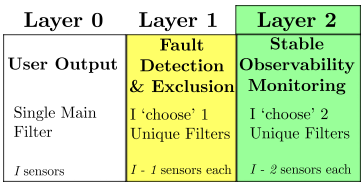


Fig. 3. ARMAS Monitoring Mode with User Output Layer 0, Fault Detection and Exclusion Sub-Layer 1, and Observability Sub-Layer 2 for Stable Observability Monitoring (SOM)

As a result of the uncorrupted subfilter guarantee provided by the ARMAS framework for a single simultaneous fault (within a monitoring epoch), SAARM is able to provide a similar guarantee for all-source position integrity. The previously stated fault exclusion axiom is extended: *assuming at least one of the subfilters is informed entirely by properly modeled, uncorrupted sensors, then at least one subfilter contains consistent state estimation error statistics* [8]. This means that the probabilistic union of the position covariance estimates of all subfilters contains the true navigation state within the statistical significance of the fault detection tests, resulting in a protection level of 0.95.

In summary, SAARM provides all-source sensor FDE and integrity for three different types of serial sensor faults. To provide resiliency to a single fault, ARMAS is required to instantiate and maintain a quantity of subfilters equal to the quantity of all-source sensors. A separate main filter is maintained strictly for user output. Fault identification is based on a sequence of $\chi^*$ statistical tests of pre-update measurement residuals. Fault exclusion is based on a subfilter consensus approach. Lastly, SAARM provides a method for all-source position integrity via the union of all subfilter position covariance estimates. This integrity concept is based on the assumption that the framework is able to maintain at least one uncorrupted subfilter.

## B. Stable Observability Monitoring (SOM)

The previous section summarized the ARMAS framework and the SAARM algorithm used to perform fault detection, exclusion, and integrity functions. The framework requires overlapping state observability to perform each of these functions. As previously stated, the framework's basic goal is to maintain at least one subfilter which is never corrupted by a faulty sensor. This is required to maintain navigation estimation consistency and provide a method for all-source integrity. The following section describes a method to accomplish this within the monitoring mode of ARMAS. Clearly, understanding the process noise, measurement noise, and measurement model are key to assessing the observability of a system state variable.

SOM provides a measure of state estimate observability in the form of a scalar and is used to directly quantify the observability of a state-variable. SOM augments SAARM and resides in the 'M' monitoring mode block in Figure 2. Since the ARMAS framework assumes state observability for both FDE and integrity, it is critical that a method for monitoring real-time position state observability is implemented prior to FDE.

SAARM requires overlapping position observability across all layer 1 subfilters to perform consistent FDE operations and guarantee the preservation of at least one uncorrupted subfilter for position integrity. For SAARM to guarantee position state observability at the layer 1 subfilter level, an additional layer of subfilters is required (Figure 3). The purpose of this layer is to provide a means for observability analysis one layer deeper than the decision-making FDE layer to maintain resiliency to a single simultaneous sensor fault (within a single monitoring epoch). The uncorrupted subfilter guarantee provided by the ARMAS framework for a single simultaneous fault enables SAARM to extend a similar guarantee for all-source position integrity [8]. SOM allows following axiom:

***If the states of interest in each layer 2 subfilter are observable and stabilizable, then each layer 1 subfilter inherits these properties.***

This is so because the layer 2 subfilters only contain a subset of sensor information available to the layer 1 subfilters. In other words, the layer 1 subfilter with the least observability and stabilizability will always be better off than the worst layer 2 subfilter.

SOM records and monitor the post-update position covariances in each $n = [1...N]$ layer 2 subfilter in the observability bank. An observability flag $O_k$ is set for layer 2 subfilter $n$ for $t_k$ according to

$$O_{k,pos}(n) = \begin{cases} 1, & \text{if } tr\big(\mathbf{P}^{[n]}_{pos}(t_k^+)\big) > tr\big(\mathbf{P}_{pos,max}\big) \\ 0, & \text{otherwise} \end{cases} \tag{35}$$

where $\mathbf{P}^{[\mathbf{n}]}_{\mathbf{pos}}(t_k^+)$ is the most recent post-update position covariance matrix for layer 2 subfilter $n$, and $\mathbf{P}_{pos,max}$ is a user-defined limit for maximum steady-state position state estimate covariance. The trace is the sum of the diagonal elements of the matrix $\mathbf{P}_k$, which represent variances of the system state estimates. If the $tr\big(\mathbf{P}^{[n]}_{pos}(t_k^+)\big)$ converges, then the individual position estimate variances also converge. When applying (35), it is important to ensure units are identical across the grouped states.

To maintain resilience to a sensor failure, a user prompt to augment ARMAS with an additional sensor is triggered if at least a single layer 2 subfilter observability test sets to 1 (36). The newly added sensor will directly enter monitoring mode if it is considered 'trusted' or must pass through sensor validation if 'untrusted'.

$$SOM_{Collaborate} = \begin{cases} true, & \text{if } \sum_{n=1}^{N} O_k(n) > 0 \\ false, & \text{otherwise} \end{cases} \tag{36}$$

where $N$ is the quantity of layer 2 subfilters.

Once a new sensor is successfully added into ARMAS monitoring mode, each layer 2 subfilter gains another sensor.

**Algorithm 1** Offboard Collaboration Pseudocode

```
if Σ_1^N [O_{k,pos}(n)] > 0 then
    SOM_Collaborate = true

    if monitoringEpoch.complete = true then
     Offboard Sensor.validate()

        if Offboard Sensor.trusted = true then
            MonitoringMode.add(Offboard Sensor)
            Repopulate Subfilter Layer_1
            Repopulate Subfilter Layer_2
            Repopulate Main Filter

        end if
    else if monitoringEpoch.complete = false then
      Offboard Sensor.gatherResiduals()
    end if
else if Σ_1^N [O_{k,pos}(n)] = 0 then
    SOM_Collaborate = false

end if
```

## C. Estimator Credibility Analysis

Estimator performance is always dependent on available measurements [17]. Accordingly, estimators should be compared using the same data set which is sufficiently large to provide statistically significant results. At first glance, position mean-squared error (MSE) appears to be a simple, practical performance metric to compare different estimators. For $n$ total measurements, the bias $\tilde{\boldsymbol{x}}$ of state estimate $\hat{\boldsymbol{x}}$ at time $i$ is shown by (37). The MSE of $\tilde{\boldsymbol{x}}$ is shown by (38).

$$\tilde{\boldsymbol{x}}_i = \boldsymbol{x}_i - \hat{\boldsymbol{x}}_i \tag{37}$$

$$MSE = \frac{1}{n}\sum_{i}^{n} \tilde{\boldsymbol{x}}_{\boldsymbol{i}}{}^2 \tag{38}$$

Although easily understandable, this definition of MSE only provides information about the first moment of estimation error. Estimators also maintain a self-assessment of the second moment of estimation error, known as the error covariance $\boldsymbol{P}$. The Normalized Estimation Error Squared (NEES) metric, $\boldsymbol{\epsilon}_i$, is Chi-Square distributed, assumes Gaussian errors, and provides a quantitative assessment of the estimator's credibility from pessimistic to optimistic [21]. NEES is calculated according to Eq. 39.

$$\boldsymbol{\epsilon}_i = (\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i)^T \boldsymbol{P}_i^{-1} (\boldsymbol{x}_i - \hat{\boldsymbol{x}}_i) \tag{39}$$

For an $m$-dimensional state estimate with $n$ total measurements, the Average NEES is shown by Eq. 40.

$$\bar{\epsilon} = \frac{1}{nm} \sum_{i=1}^{n} \boldsymbol{\epsilon}_i \tag{40}$$

NEES represents the square of the distance between the state estimate $\hat{\boldsymbol{x}}$ and the truth $\boldsymbol{x}$, normalized by the error covariance $\boldsymbol{P}$. Average NEES can be interpreted as a unidimensional average of this squared distance. Average NEES quantifies estimator credibility by assessing the accuracy of employed assumptions [21]. If the Average NEES is near 1, then the estimator is likely credible. If the Average NEES is much less (greater) than 1, it is pessimistic (optimistic).

## IV. Methods for Managing Cross-Correlated Information

We define a collaborative exchange as a transaction between a single requesting recipient and a single cooperative donor. We must be careful when fusing collaborative sensor data because it can result in dependency between state estimates and measurements if a relative measurement is later re-used without taking historical dependencies into account. This problem is called double-counting [22]. Distributed networks are scalable and resilient to changes in topology but are particularly vulnerable to this problem.

Additionally, sensor corruption is likely detected *a posteriori*, especially in distributed networks. The reason for this is that the source of the corruption is often unknown until the distributed statistical tests detect it. At this point, a method to decorrelate corrupted information is required to preserve consistency. Since Bayesian estimation incorporates historical measurements, we must maintain a fault-resistant architecture which contains estimators never corrupted to eliminate the corrupted historical dependencies and recover consistency.

There are known approaches to manage the double-counting problem and the *a posteriori* consistency recovery problems. The following section analyzes the Covariance Intersection and the distributed Interleaved Update Algorithm. The section concludes with an introduction to the novel Split Approach designed to manage cross-correlation in a fault-resistant collaborative network.

### 1. Covariance Intersection Method

The limitations of managing unknown cross-correlation between state estimates and measurements in decentralized estimation with arbitrary network topologies are widely known. A rudimentary fix for this involves increasing Bayesian estimator process noise to account for unmodelled dependencies between measurements and the state estimate. This strategy leads to increased state estimate covariance and degradation in overall estimator performance. This motivated the CI method [22][23] which fuses the probability densities of two measurements with unknown cross-correlation into a consistent covariance estimate. The CI technique is recommended as a good estimator for highly cross-correlated measurements. The CI method utilizes a conservative estimate of $P_{ab}$ to guarantee estimator consistency. Although the CI method is a conservative solution for the double-counting problem, a primary downside for our application is the inability to recover previously fused information in the event of a sensor corruption discovered *a posteriori*.

### 2. Interleaved Update (IU) Algorithm

The IU algorithm [24] provides a multi-filter approach to the inconsistency problem. This algorithm retains consistency by maintaining separable state estimates $\mathbf{X_i}$ and covariance estimates $\mathbf{P_i}$ for collaborative offboard measurements. For example, the covariance matrices associated with vehicles $i, j$ are assembled in a block diagonal matrix $\mathbf{P_{i,j}}(t)$ as shown in (41).

$$\mathbf{P_{i,j}}(t) = \left[\begin{array}{c|c} P_i(t) & 0 \\ \hline 0 & P_j(t) \end{array}\right] \tag{41}$$

The timestamp of the most recent measurement update from each vehicle is maintained in a $\mathbf{T_i}$ matrix where each row $q$ represents a filter and each column $i$ represents an offboard vehicle number. Row 1 in $\mathbf{T_i}$ corresponds to the set of states which has never been updated by an offboard range measurement. All filters are always updated by onboard measurements.

Each time a collaborative measurement is broadcast from vehicle $i$ to another vehicle, it contains current copies of $\mathbf{X_i}$, $\mathbf{P_i}$, and $\mathbf{T_i}$. With these parameters, the receiving vehicle $j$ always has access to a state estimate and covariance which have only been updated by sensors on the transmitting vehicle $i$. This uncorrelated information can be used by vehicle $j$ without inconsistency.

A vehicle $i$ receives a broadcast from vehicle $j$ containing information $\mathbf{X_j}$, $\mathbf{P_j}$, and $\mathbf{T_j}$ to update filter $q$ onboard vehicle $i$. If information from vehicles other than $j$ is present, the optimal combination of filter information is selected for fusion by examining the trace of the fused covariance estimate $\mathbf{P_i^q}$. The combination of filter information that results in the $argmin\big[trace(\mathbf{P_i^q})\big]$ is used to update filter $q$ onboard vehicle $i$.

The IU algorithm requires each vehicle to maintain $2^n$ simultaneous Kalman filters. The author speculated that 30 unique vehicle IDs are attainable with this framework [24]. Exponential scalability is highly undesirable, especially for multi-filter residual monitoring algorithms like ARMAS [25]. Even so, the idea of retaining a proprioceptive solution uncorrupted by offboard collaboration is a great insurance policy which enables eventual recovery to autonomy from collaboration.

### 3. Split Approach (aka Stingy Collaboration)

If each vehicle maintains two parallel instances of ARMAS-SOM, one collaborative and the other proprioceptive, then we can completely eliminate double-counting and simultaneously mitigate the possibility of

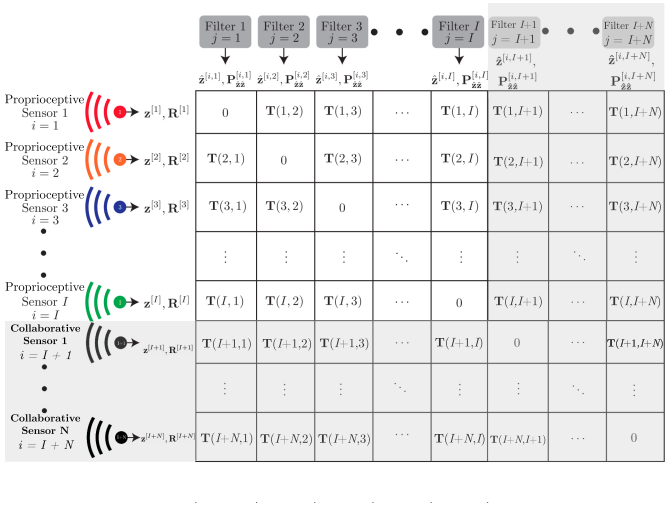


Fig. 4. Collaborative ARMAS T-matrix: Collaborative information (gray) is used to augment proprioceptive information (white). This forms the T-matrix structure for the collaborative all-source framework which is fused into a single main filter for user output. A parallel proprioceptive instance is maintained separately.

propagating corrupt donor data across the network. For example, if a single unknowingly corrupted donor shares their proprioceptive navigation solution with a recipient, the collaborative instance of ARMAS-SOM on the recipient's vehicle already propagates a subfilter which excluded any measurements from the corrupted donor.

For example, a single wingman in a 5-vehicle collaborative network unknowingly broadcasts corrupted proprioceptive donor information to all vehicles Since each wingman's collaborative estimation framework is propagating a subfilter "No Wingman 3" which excludes any measurements from Wingman 3, a single FDE event is required to remove all historical contributions from Wingman 3. Throughout this process, Wingman 3 is able to receive consistent proprioceptive donor information from the other networked vehicles which facilitates FDE actions onboard Wingman 3 to recover from the fault.

With the stabilization guarantee afforded by SOM, the ARMAS-SOM framework will detect the faulty donor and exclude them from the donor's navigation solution [14]. In other words, a single FDE action is required by the recipient to eliminate all inconsistencies resulting from the donor. Although this approach does not glean the maximum quantity of information from the network, it significantly maximizes efficent use of network resources, lowers the risk of repeatedly propagating inconsistent information, and provides a mechanism to eliminate the faulty information completely.

If more than one donor vehicle is corrupted (i.e. we violate the single simultaneous failure assumption), the recipient vehicle can revert to the proprioceptive instance of ARMAS-SOM which completely eliminates any possibility of offboard corruption. This is particularly important because ARMAS-SOM must always maintain at least one uncorrupted subfilter to guarantee consistency. Figure 4 shows the structural similarities between the T-matrices used in the proprioceptive and collaborative framework instances. If the donors shared collaborative solutions which contained contributions from other corrupted donor vehicles, an exponentially scaled approach like the IU Algorithm above would be required to back-track and sort out how to fuse the information without loss of consistency. This split approach is useful as an insurance policy to regain autonomous navigation by eliminating all contributions from other vehicles.

## V. Guidelines for Collaborative All-Source Navigation

We offer the following guidelines for collaborative all-source navigation:

- The key to integrity is the ability to maintain at least one consistent estimator.
- Inconsistency can result from on-board corruption, off-board corruption, or by double-counting off-board information.
- Employ a scalable architecture to fuse collaborative information in a manner which facilitates the recovery of consistency after fault detection.
- A proprioceptive filter is a great insurance policy.

## VI. Simulation

Consider a three-dimensional example in a local tangential (navigation) frame with five independent distributed "wingmen" air vehicles at different formation location, each operating the ARMAS-SOM framework. Each vehicle maintains split, parallel autonomous (proprioceptive) and collaborative instances of ARMAS-SOM propagating 10-state EKFs for position, velocity, acceleration, and GNSS clock bias states. The kinematics block is defined by

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{\mathbf{x}}_p(t) \\ \dot{\mathbf{x}}_v(t) \\ \dot{\mathbf{x}}_a(t) \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{x}}_v(t) \\ \dot{\mathbf{x}}_a(t) \\ -\frac{1}{\tau_a}\dot{\mathbf{x}}_v(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \mathbf{w}(t) \end{bmatrix} \tag{42}$$

where $\mathbf{x}_p$ is the vehicle's position (m), $\mathbf{x}_v$ is the vehicle's velocity (m/s), $\mathbf{x}_a$ is the vehicle's acceleration (m/s$^2$), and $\tau_a = 300$ seconds is a time-constant associated with a First-order Gauss-Markov (FOGM) process. A 3D white noise process is given by $w(t)$ where E[$\mathbf{w}$(t) $\mathbf{w}$(t+$\tau$)$^T$] = $\mathbf{Q}\delta(\tau)$ and

$$\mathbf{Q} = (1.0 \times 10^{-2})^2 \mathbf{I}_{3\times 3}\ (m^2/s^4) \tag{43}$$

Each vehicle is initialized with unique initial positions according to

$$\hat{\mathbf{x}}_1(0) = \begin{bmatrix} 0 & 0 & 3000 \end{bmatrix}^T \text{ meters,}$$

$$\hat{\mathbf{x}}_2(0) = \begin{bmatrix} -10000 & 3000 & 5000 \end{bmatrix}^T \text{ meters,}$$

$$\hat{\mathbf{x}}_3(0) = \begin{bmatrix} 10000 & 3000 & 5000 \end{bmatrix}^T \text{ meters,}$$

$$\hat{\mathbf{x}}_4(0) = \begin{bmatrix} -10000 & -3000 & 2000 \end{bmatrix}^T \text{ meters,}$$

$$\hat{\mathbf{x}}_5(0) = \begin{bmatrix} 10000 & -3000 & 2000 \end{bmatrix}^T \text{ meters.}$$

Each vehicle is initialized with identical state error covariance matrices according to

$$\mathbf{P}_n(0) = diag(\begin{bmatrix} 20^2 & 20^2 & 20^2 & 10^2 & 10^2 & 10^2 & 0.01^2 & 0.01^2 & 0.01^2 & 8000^2 \end{bmatrix}). \tag{44}$$

Each aircraft receives discrete measurements from a constellation of 5 stationary satellite vehicles (SV). The constellation is uniformly distributed in azimuth and simulates favorable coverage with elevation angles between approximately 45 degrees and 63.4 degrees. Each vehicle is equipped with dual-frequency GNSS receivers receiving simulated pseudorange measurements from 5xL1 (1575 MHz), and 5xL2 (1227 MHz) SVs.

Individual pseudorange measurements are performed according to

$$\rho_i = \sqrt{(X_{SV,i} - X_u)^2 + (Y_{SV,i} - Y_u)^2 + (Z_{SV,i} - Z_u)^2} + b_u \tag{45}$$

where $\rho_i$ is the pseudorange to SV $i$, with fixed coordinates $(X_{SV}, Y_{SV}, Z_{SV})$, estimated user coordinates are $(X_u, Y_u, Z_u)$, and an estimated GNSS receiver clock bias is $b_u$. The pseudorange measurement covariance is $\mathbf{R_{SV}}$ = $10^2$ m$^2$. A receiver clock bias $b_u$ is independently estimated as an additional state in each EKF. Each vehicle is able to augment with collaborative ranging measurements from nearby wingmen according to

$$R_j = \sqrt{(X_{wng,j} - X_r)^2 + (Y_{wng,j} - Y_r)^2 + (Z_{wng,j} - Z_r)^2}. \tag{46}$$

where $R_j$ is a simulated radar range to Wingman $j$ with actual coordinates $(X_{wng}, Y_{wng}, Z_{wng})$ and with actual recipient coordinates $(X_r, Y_r, Z_r)$. Unlike the GNSS sensors, no ephemeris is available, so estimated recipient and wingman solutions are fused with (46) to form the measurement update. The main solution from the user's collaborative instance provides the estimated user location and covariance. The wingman's proprioceptive instance provides estimated wingman location and position covariance. The ranging measurement covariance is $\mathbf{R}_{WING}$ = $10^2$ m$^2$. Collaborative updates are performed only in the collaborative instance for each recipient. Clock synchronization is assumed and sensor delays are assumed to be minimal. Vehicles are equipped with simulated alternative visual navigation sensors which provide noisy velocity updates with $\mathbf{R}_{VIS}$ = $(10^2)\mathbf{I}_{3\times3}$ (m/s)$^2$. All sensors provide measurement updates at 1Hz. SOM *a posteriori* state estimate covariance parameters are set to $\beta$ = 5 and $\mathbf{P}_{j,max}$ = $(25\text{m})^2 \times 3$ = 6889m$^2$. The vehicles travel at approximately 75 meters/second in the $+Y$ direction in a LTF for a total of 27 minutes (1620 seconds) of run time.

The simulation is divided into 4 separate regions (Figure 5):

1) Permissive (0-5km)
2) L1 Jamming (5-50km)
3) L1 Jamming + L2 Spoofing (50km-100km)
4) Permissive (100km - End of Sim)

All five aircraft initialize with $5 \times L1$ trusted pseudorange sensors, $5 \times L2$ trusted pseudorange sensors, and trusted

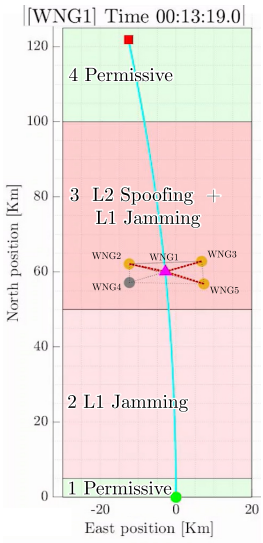


Fig. 5. Overhead view of GNSS Regions with WNG1 in center of formation at T = 13m 19s (799s). Wingman locations are: WNG2 (Upper Left), WNG3 (Upper Right), WNG4 (Lower Left), WNG5 (Lower Right)

$1 \times VIS$ sensor in region 1 at $T = 0$ seconds. Once "WNG1" enters region 2 at $T = 60$ seconds, all vehicles encounter $L1$ noise jamming and pare down to $5 \times L2$ pseudorange sensors and $1 \times VIS$ sensor.

In region 2, ARMAS-SOM detects a threat to resilience and each vehicle selectively validates untrusted individual offboard collaborators until the observability warnings are rescinded. Each donor vehicle shares its proprioceptive solution according to the split "stingy collaboration" approach described in section 3 and each recipient vehicle performs a collection period to ensure measurement residuals are valid for acceptance. Each vehicle augments with approximately 2-3 collaborative relationships for prior to entering region 3. Once "WNG1" enters region 3 at $T = 660$ seconds, all vehicles experience an insidious pseudorange bias on $L2SAT4$ which initializes at 10 meters and grows at 10 m/s.

Once each vehicle detects the growing pseudorange bias and determines the culprit, $L2SAT4$ is excluded from each ARMAS-SOM instance. In response to the $L2SAT4$ exclusion, ARMAS-SOM detects a threat to resilience and forms an additional collaborative relationship to stabilize (See Figure 6). The vehicles transition into region 4 at $T = 1320$ seconds where $L1$ jamming and $L2$ spoofing end. Once each vehicle's proprioceptive ARMAS-SOM instances re-establish resilience, then the collaborative relationships are removed and the vehicles gracefully regain autonomy.

## VII. Numerical Results

The results of the first simulation show that ARMAR-SOM's stingy collaboration scheme stabilizes the all-source Guaranteed Position Zone (GPZ) during jamming and spoofing events as shown in Figure 6. A comparison

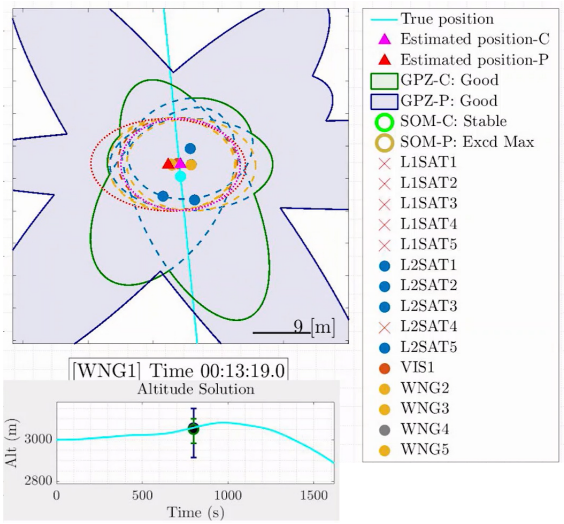


Fig. 6. Combined Overhead and Profile View of WNG1 Split Collaborative "GPZ-C" and Proprioceptive "GPZ-P" (PL = 0.95) ARMAS-SOM Instances with individual Layer 1 Subfilter Solutions at T = 13m 19s (T = 799s) inside the L1 Jamming + L2 Spoofing Region 3

of the GPZ dimensions for both the proprioceptive and collaborative ARMAS-SOM instances is shown in Figure 7. This shows that the SOM algorithm sufficiently augments itself using collaboration to stabilize the all-source position integrity solution during the combined navigation anomalies. Since the GPZ (Protection Level = 0.95) is a visualization of the union of the FDE layer solutions, this is proof that observability layer stabilization is inherited by the FDE and main layer. The end result of this contribution is guaranteed resilience to a single simultaneous failure for 81% of the total simulation time versus only 5% for a non-collaborative ARMAS instance. A detailed derivation of the integrity metrics used to develop the GPZ is provided by [8].

Figure 8 shows estimator credibility results in the form of NEES for each wingman. The mean results for 3D-RSS Error and NEES are visible in Table I. This shows that "stingy" collaboration preserves consistent estimation while mititgating the risk of undesired error propagation and avoiding the double-counting problem normally associated with unknown cross-correlation.

| Instance | Veh. 1 | Veh. 2 | Veh. 3 | Veh. 4 | Veh. 5 | Grand Mean |
|---|---|---|---|---|---|---|
| *Proprioceptive Mean RSS Error (m)* | 9.86 | 13.39 | 9.90 | 11.13 | 9.52 | 10.76 |
| *"Stingy" Collaborative Mean RSS Error (m)* | 8.89 | 10.39 | 8.37 | 10.82 | 9.22 | 9.54 |
| *Proprioceptive Avg. NEES* | 0.94 | 1.16 | 0.74 | 0.98 | 0.78 | 0.92 |
| *"Stingy" Collaborative Avg. NEES* | 0.88 | 1.04 | 0.69 | 1.10 | 0.81 | 0.90 |

TABLE I
Comparison of Proprioceptive and Collaborative ARMAS-SOM Instances

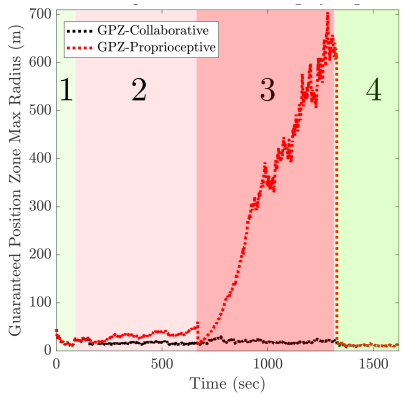


Fig. 7. Size Comparison of Split "Stingy" Collaborative and Proprioceptive All-source Guaranteed Position Zones for WNG1 in GNSS Regions 1-4

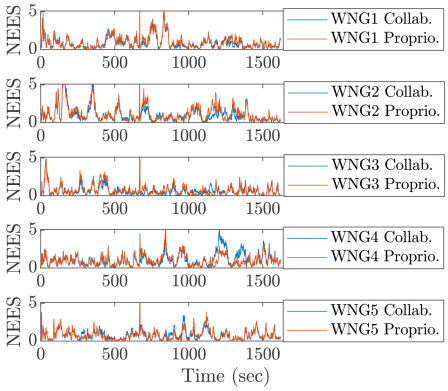


Fig. 8. Estimator Credibility Comparison for Split Collaborative and Proprioceptive Instances of ARMAS-SOM for 5 Vehicles

## VIII. Conclusion

This research presented the motivation for collaborative all-source navigation and covered a brief background on the fundamental concepts required to understand the resilient ARMAS-SOM framework. We explained how SOM provides intrinsic information awareness for timely selective collaboration to preserve estimator consistency with the assumption of a single simultaneous sensor failure. We introduced a novel split structure which maintains both "stingy" collaborative and proprioceptive frameworks to maximize efficient use of network resources, limit the propagation of unknown corruption to a single donor, donor prioritization, and eliminate double counting normally associated with recursive Bayesian fusion of collaborative information. We also demonstrated how the same intrinsic information awareness used by SOM can be used to determine when it is safe to transition back to autonomy from collaboration. This structure demonstrated effective stabilization of the all-source GPZ integrity solution, a 11% reduction in RSS error, preservation of consistent estimation, and the ability to gracefully recover autonomy. Follow on work in this area involves lab testing with real sensors with the eventual goal of flight test.

## References


[1] Michael J. Veth. "Fusion of Imaging and Inertial Sensors for Navigation". PhD thesis. Air Force Institute of Technology, 2006.

[2] Joseph Curro and John Raquet. "Navigation using VLF environmental features". In: *Position, Location and Navigation Symposium (PLANS), 2016 IEEE/ION*. IEEE. 2016, pp. 373–379.

[3] Aaron Canciani and John Raquet. "Absolute positioning using the Earth's magnetic anomaly field". In: *Navigation* 63.2 (2016), pp. 111–126.

[4] Kenneth Fisher and John Raquet. "Precision Position, Navigation, and Timing without the Global Positioning System". In: *Navigation* 2 (2010), pp. 24–44.

[5] John Raquet and Richard Martin. "NON-GNSS RADIO FREQUENCY NAVIGATION". In: *Signals* (2008), pp. 5308–5311.

[6] Ramsey Faragher and Robert Harle. "Location fingerprinting with bluetooth low energy beacons". In: *IEEE Journal on Selected Areas in Communications* 33.11 (2015), pp. 2418–2428. ISSN: 07338716. DOI: 10.1109/JSAC.2015.2430281.

[7] Dr. Dave Temper. *DARPA Adaptable Navigation Systems (ANS)*. 2010. URL: https://www.darpa.mil/program/adaptable-navigation-systems (visited on 04/06/2020).

[8] Juan Jurado et al. "Residual-Based Multi-Filter Methodology for All-Source Fault Detection, Exclusion, and Performance Monitoring". In: *NAVIGATION* (2019).

[9] Juan Blanch et al. "Baseline advanced RAIM user algorithm and possible improvements". In: *IEEE Transactions on Aerospace and Electronic Systems* 51.1 (2015), pp. 713–732. ISSN: 00189251. DOI: 10.1109/TAES.2014.130739.

[10] Mathieu Joerger and Boris Pervan. "Kalman Filter-Based Integrity Monitoring Against Sensor Faults". In: *Journal of Guidance, Control, and Dynamics* 36.2 (2013), pp. 349–361. DOI: 10.2514/1.59480. eprint: https://doi.org/10.2514/1.59480. URL: https://doi.org/10.2514/1.59480.

[11] Çağatay Tanıl, Samer Khanafseh, and Boris Pervan. "Detecting Global Navigation Satellite System Spoofing Using Inertial Sensing of Aircraft Disturbance". In: *Journal of Guidance, Control, and Dynamics* 40.8 (2017), pp. 2006–2016. DOI: 10.2514/1.G002547. eprint: https://doi.org/10.2514/1.G002547. URL: https://doi.org/10.2514/1.G002547.

[12] Mathieu Joerger and Boris Pervan. "Fault detection and exclusion using solution separation and chi-squared ARAIM". In: *IEEE Transactions on Aerospace and Electronic Systems* 52 (2 Apr. 2016), pp. 726–742. ISSN: 00189251. DOI: 10.1109/TAES.2015.140589.

[13] Juan Jurado, John F. Raquet, and Christine M. Schubert-Kabban. "Autonomous and Resilient Management of All-source sensors for Navigation". In: *ION* (2018).

[14] Jonathon S Gipson and Robert C Leishman. "Resilience for Multi-filter All-source Navigation Framework with Integrity". In: *AFIT Faculty Preprint* (2021).

[15] R. E. Kalman. "A New Approach to Linear Filtering and Prediction Problems". In: *Journal of Basic Engineering* 82.1 (1960), p. 35. DOI: 10.1115/1.3662552. URL: https://doi.org/10.1115%2F1.3662552.

[16] Charles F. Van Loan. "Computing Integrals Involving the Matrix Exponential". In: *IEEE Transactions on Automatic Control*. Vol. 23:3. June 1978, pp. 395–404.

[17] Peter S. Maybeck. *Stochastic Models, Estimation, and Control Volume 1*. Virginia: Navtech, 1982.

[18] Kyle Kauffman et al. "Scorpion : A Modular Sensor Fusion Approach for Complementary Navigation Sensors". In: *ION/IEEE PLANS*. Portland, OR: IEEE, 2020.

[19] Fredric M Ham. "Observability, Eigenvalues, and Kalman Filtering". In: *Ieee Transactions On Aerospace And Electronic Systems* 2 (1983), pp. 269–273.

[20] Roy De Maesschalck, Delphine Jouan-Rimbaud, and Désiré L Massart. "The mahalanobis distance". In: *Chemometrics and intelligent laboratory systems* 50.1 (2000), pp. 1–18.

[21] Zhaozhong Chen et al. "Weak in the NEES?: Auto-Tuning Kalman Filters with Bayesian Optimization". In: *2018 21st International Conference on Information Fusion, FUSION 2018* (2018), pp. 1072–1079. DOI: 10.23919/ICIF.2018.8454982. arXiv: 1807.08855.

[22] Simon J. Julier and Jeffrey K. Uhlmann. "Non-divergent estimation algorithm in the presence of unknown correlations". In: *Proceedings of the American Control Conference* 4 (1997), pp. 2369–2373. ISSN: 07431619. DOI: 10.1109/acc.1997.609105.

[23] S. J. Julier. "Fusion without independence". In: *IET Seminar Digest* 2008.12273 (2008), pp. 1–4. DOI: 10.1049/ic:20080050.

[24] Alexander Bahr, Matthew R. Walter, and John J. Leonard. "Consistent cooperative localization". In: *Proceedings - IEEE International Conference on Robotics and Automation* (2009), pp. 3415–3422. ISSN: 10504729. DOI: 10.1109/ROBOT.2009.5152859.

[25] Louis Sepulveda et al. "Optimizing a Bank of Kalman Filters for Navigation Integrity using Efficient Software Design." In: *Proceedings of the ION GNSS+* (2021).

**Dr. Jonathon Gipson** is currently the Deputy Director of Education at the USAF Test Pilot School (TPS), Edwards AFB, CA. He also serves as an Assistant Professor and Instructor Flight Test Engineer. He is a graduate of the Rose-Hulman Institute of Technology (RHIT), Terre Haute, IN. He earned his Ph.D. in 2021 from the Air Force Institute of Technology (AFIT) working under the Autonomy and Navigation Technology (ANT) center. His interests include collaboration, sensor fusion, and mission systems integration.

**Dr. Rob Leishman** is currently the Director of the Autonomy and Navigation Technology (ANT) Center at the Air Force Institute of Technology (AFIT), which he began in April 2019. He is also a Research Assistant Professor in the Department of Electrical and Computer Engineering at AFIT. Prior to AFIT, Dr. Leishman worked at the Air Force Research Laboratory. He earned the Ph.D. in 2013 in Mechanical Engineering at Brigham Young University in Provo, UT through support from the DoD SMART Scholarship. His research interests include autonomous vehicles, robust alternative navigation, image processing, sensor fusion, and control.