AN ENHANCED FRAMEWORK FOR COLLABORATIVE ALL-SOURCE NAVIGATION WITH INTEGRITY

DISSERTATION

Jonathon S. Gipson, Major, USAF

AFIT-ENV-DS-21-S-065

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

The views expressed in this dissertation are those of the author and do not reflect the official policy or position of the United States Air Force, the Department of Defense, or the United States Government.

This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENV-DS-21-S-065

AN ENHANCED FRAMEWORK FOR COLLABORATIVE ALL-SOURCE NAVIGATION WITH INTEGRITY

DISSERTATION

Presented to the Faculty

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

Jonathon S. Gipson, B.S.E.E., M.B.A., M.S.E.E, M.S.FTE

Major, USAF

2021

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

AFIT-ENV-DS-S-065

AN ENHANCED FRAMEWORK FOR COLLABORATIVE ALL-SOURCE NAVIGATION WITH INTEGRITY

Jonathon S. Gipson, B.S.E.E., M.B.A., M.S.E.E, M.S.FTE
Major, USAF

Approved:

LEISHMAN.ROBERT.C.1292647626 Digitally signed by LEISHMAN.ROBERT.C.1292647626 Date: 2021.08.11 09:49:53 -04'00'

Robert C. Leishman, Ph.D. (Chairman) — 11 Aug 2021 Date

COLOMBI.JOHN.MICHAEL.1010650646 Digitally signed by COLOMBI.JOHN.MICHAEL.1010650646 Date: 2021.08.08 21:22:32 -06'00'

John M. Colombi, Ph.D. (Member) — 8 Aug 2021 Date

SCHUBERT KABBAN.CHRISTINE.M.1283160374 Digitally signed by SCHUBERT KABBAN.CHRISTINE.M.1283160374 Date: 2021.08.12 09:25:11 -04'00'

Christine M. Schubert-Kabban, Ph.D. (Member) — 12 Aug 2021 Date

Accepted:

BADIRU.ADEDEJI.B.1293329580 Digitally signed by BADIRU.ADEDEJI.B.1293329580 Date: 2021.08.13 12:05:08 -04'00'

Adedeji B. Badiru, Ph.D. — 13 Aug 2021 Date

Dean, Graduate School of Engineering and Management

Abstract

The Autonomous and Resilient Management of All-source Sensors with Stable Observability Monitoring (ARMAS-SOM) framework fuses collaborative all-source sensor information in a resilient manner with fault detection, exclusion, and integrity solutions recognizable to a Global Navigation Satellite System (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 *a posteriori* 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 instances of ARMAS-SOM 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.

To my wife and kids, thank you!

Acknowledgments

Thanks go to my family, my advisor, and my committee for making this possible.

Jonathon S. Gipson

Table of Contents

Page

Abstract iv

Dedication v

Acknowledgments vi

Table of Contents vii

List of Figures xi

List of Tables xvii

List of Acronyms xviii

I. Introduction 1

1.1 Summary of Related Research 2
1.1.1 Vehicle Guidance for Self-Localization 2
1.1.2 Cooperative Navigation Support 2
1.1.3 Observability Monitoring 3
1.1.4 Collaborative Navigation 3
1.2 Research Contributions 4
1.3 Outline 6

II. Background 8

2.1 Notation Conventions 8
2.1.1 Reference Frames 9
2.2 The All-Source Navigation Problem 12
2.2.1 Recursive Bayesian Estimation 12
2.2.2 Basic Residual Monitoring 14
2.2.3 Inertial Navigation 15
2.2.4 All-Source Navigation 16
2.3 Autonomous Resilient Management of All-source Sensors (ARMAS) . . . 18
2.3.1 Sensor-Agnostic All-source Residual Monitoring (SAARM) 22
2.4 Observability Assessment Methods 27
2.4.1 Eigenvalue Normalization for the Error Covariance Matrix 29
2.4.2 Degree of Observability 30

Page

2.5 Methods for Managing Cross-Correlated Information . . . 31
2.5.1 Covariance Intersection Method . . . 32
2.5.2 Ignore Cross-Correlation Technique (Conditional Naive Filter) . . . 34
2.5.3 Game-Theoretic Filter (GTF) . . . 37
2.5.4 Interleaved Update (IU) Algorithm . . . 37
2.5.5 Collaborative Localization Simulation . . . 39
2.5.6 Estimator Credibility Analysis . . . 41
2.5.7 Numerical Results . . . 42
2.5.8 Collaborative Resiliency Against Simultaneous Sensor Failures . . . 44
2.5.9 Chapter Summary . . . 49

III. Real-time Trajectory Optimization for Collaborative Self-Localization in Random Aircraft Formations . . . 50

3.1 Introduction . . . 50
3.2 Self-Aligning Swarm (SAS) Algorithm for Self-localization . . . 51
3.2.1 Background . . . 51
3.2.2 2-D CRLB for Range Measurements in AWGN . . . 52
3.2.3 Pseudolite Cofactor Matrix and SAS Cost Function . . . 57
3.2.4 Simulation . . . 58
3.3 Numerical Results . . . 65
3.4 Chapter Summary and Future Work . . . 66

IV. Swarm Control for Autonomous Navigation Support . . . 70

4.1 Introduction . . . 70
4.2 Related Work . . . 73
4.2.1 Decentralized-Local vs. Centralized-Global Optimization . . . 73
4.2.2 Magnetic Navigation . . . 74
4.3 Swarm Control for Autonomous Navigation Support (SCANS) Algorithm for Swarm Control . . . 76
4.3.1 Introduction . . . 76
4.3.2 Localization Algorithm . . . 76
4.3.3 Pluggable Cost Function . . . 78
4.3.4 Simulation . . . 80
4.4 Numerical Results . . . 90
4.5 Large Swarm Behavior . . . 90
4.6 Chapter Summary and Future Work . . . 97

V. Stable Observability Monitoring . . . 105

5.1 Introduction . . . 106

Page

5.2 Background . . . 106
5.2.1 Autonomous Resilient Management of All-source Sensors (ARMAS) . . . 106
5.2.2 Sensor-Agnostic All-source Residual Monitoring (SAARM) . . . 109
5.2.3 Motivation for Stable Observability Monitoring (SOM) . . . 115
5.3 Novel Observability Layer 2 "Sub-subfilter" Bank . . . 120
5.4 Stable Observability Monitoring (SOM) . . . 122
5.5 State Estimate Covariance Transient Growth Threshold (Beta) and Maximum State Estimate Covariance Limits . . . 125
5.6 Simulation . . . 127
5.7 Numerical Results . . . 129
5.7.1 Scenario 1 . . . 130
5.7.2 Scenario 2 . . . 133
5.7.3 Scenario 3 . . . 136
5.7.4 Scenario 4 . . . 139
5.7.5 Across Scenarios . . . 142
5.8 Chapter Summary and Future Work . . . 142

VI. A Framework for Resilient Collaborative All-source Navigation . . . 148

6.1 Introduction . . . 148
6.2 Background . . . 150
6.2.1 Recursive Bayesian Estimation . . . 150
6.2.2 Residual Monitoring . . . 152
6.3 Autonomous Resilient Management of All-source Sensors (ARMAS) . . . 153
6.3.1 Sensor-Agnostic All-source Residual Monitoring (SAARM) . . . 156
6.3.2 Stable Observability Monitoring (SOM) . . . 162
6.3.3 Estimator Credibility Analysis . . . 164
6.4 Methods for Managing Cross-Correlated Information . . . 165
6.4.0.1 Covariance Intersection Method . . . 166
6.4.0.2 Interleaved Update (IU) Algorithm . . . 167
6.4.0.3 Split Approach (aka Stingy Collaboration) . . . 168
6.5 Guidelines for Collaborative All-Source Navigation . . . 169
6.6 Simulation . . . 169
6.7 Numerical Results . . . 173
6.8 Chapter Summary and Future Work . . . 173

VII. Summary and Conclusions . . . 182

Bibliography . . . 184

Page

Vita 191

List of Figures

Figure Page

2.1 Navigation Reference Frames 11

2.2 Aircraft Body reference frame (b-frame) [24] 12

2.3 ARMAS Framework State Diagram [43] 21

2.4 SAARM T-Matrix for $i = 1...I$ All-Source Sensors [44] 26

2.5 SAARM All-Source Horizontal Position Integrity (SCORPION Pluggable EKF) 28

2.6 Comparison of Fused Error Covariance Ellipses P_{cc} resulting from CI $tr(P_{cc})$ vs. CI $det(P_{xx})$ vs. Naive Kalman Filter Method with 2 Cross-Correlated Range Measurements 35

2.7 Comparison of Fused Error Covariance Ellipses P_{cc} resulting from CI $tr(P_{cc})$ vs. CI $det(P_{xx})$ vs. Naive Kalman Filter Method with 4 Cross-Correlated Range Measurements 36

2.8 Position Error for Collaborative Vehicles 1, 2 45

2.9 Position Error for Collaborative Vehicles 3, 4 46

2.10 Normalized Estimation Error Squared for Collaborative Vehicles 1, 2 47

2.11 Normalized Estimation Error Squared for Collaborative Vehicles 3, 4 48

3.1 Visualization of Partial Derivatives of D_n 55

3.2 Optimal 2-D Self-Localization Geometry for $N = 7$ Anchor Nodes 56

3.3 SAS Initialization with Random Node Locations and Initial Trajectory, $t_0 = 0.0$ sec 60

3.4 SAS Formation Node Trajectories with Angular Perturbations, $t = 1.0$ sec . . . 61

3.5 Initial Node Trajectory Towards Local Minimum of Cost Function, $t_0 = 0.0$ sec 63

3.6 Contour of $tr(CRLB_{\hat{\theta}})$ for Formation Node Network, $t_0 = 0.0$ sec 64

3.7 Distribution of Mean Position MSE for 10K Trials, $T_{trial} = 150$ sec, $\sigma_{range} = 5$ m 67

Figure Page

3.8 Comparison of Mean Position MSE Distributions for 10K Trials, T_{trial} = 150 sec, σ_{range} = 5 m 68

3.9 Comparison of Normal-Gaussian Models Fit to SAS-Optimized and Formation Node 1 Mean Position MSE Distributions for 10K trials, T_{trial} = 150 sec, σ_{range} = 5 m 69

4.1 Importance-Weighted Navigation Support using Surveillance Network 71

4.2 Snapshot for 10 Vehicle Network at t_i = 1.0 sec, 3 Mission Vehicles with weights of {0, 1, 10} and 7 Navigation Support Vehicles 81

4.3 Weighted Cofactor Matrix Trace Surface Snapshot for 10 Vehicle Network at t_i = 1.0 sec, 3 Mission Vehicles with weights of {0, 1, 10} and 7 Navigation Support Vehicles from Perspective of Navigation Vehicle 1 81

4.4 3-D Weighted Cofactor Matrix Trace Surface Snapshot for 10 Vehicle Network at t_i = 1.0 sec, 3 Mission Vehicles with weights of {0, 1, 10} and 7 Navigation Support Vehicles from Perspective of Navigation Vehicle 1 82

4.5 Weighted Cost Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 1 sec 82

4.6 3-D Weighted Cost Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 1 sec 83

4.7 Snapshot for Stabilized 10 Vehicle Network at t_i = 100 sec, 3 Mission Vehicles and 7 Navigation Support Vehicles 83

4.8 Weighted Cofactor Matrix Trace Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 100 sec 84

4.9 3-D Weighted Cofactor Matrix Trace Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 100 sec 85

Figure Page

4.10 Weighted Cost Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 100 sec 86
4.11 3-D Weighted Cost Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 100 sec 91
4.12 Example 2-D Position RMSE for Magnetic Navigation Support Vehicle 4, Trial 3 91
4.13 2-D Position RMSE Mission Vehicle 2 vs Mission Vehicle 3, Monte Carlo Trial 1 92
4.14 2-D Position RMSE Mission Vehicle 2 vs Mission Vehicle 3, Monte Carlo Trial 2 93
4.15 2-D Position RMSE Mission Vehicle 2 vs Mission Vehicle 3, Monte Carlo Trial 3 94
4.16 2-D Position RMSE Mission Vehicle 2 vs Mission Vehicle 3, Monte Carlo Trial 4 95
4.17 Snapshot for Initialized 50 Vehicle Network at t_i = 1 sec, 3 Mission Vehicles and 47 Navigation Support Vehicles 98
4.18 Weighted Cofactor Matrix Trace Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 1 sec 99
4.19 Weighted Cofactor Matrix Trace Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 1 sec 100
4.20 Weighted Cost Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 1 sec 101
4.21 3-D Weighted Cost Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 1 sec 102
4.22 Snapshot for Stabilized 50 Vehicle Network at t_i = 250 sec, 3 Mission Vehicles and 47 Navigation Support Vehicles 102
4.23 Weighted Cofactor Matrix Trace Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 250 sec 103
4.24 3-D Weighted Cofactor Matrix Trace Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 250 sec 103

Figure Page

4.25 Weighted Cost Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 250 sec 104

4.26 3-D Weighted Cost Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 250 sec 104

5.1 ARMAS Framework State Diagram [43]. A sensor begins at point **O** (origin) and is "trusted" or "untrusted". SAARM and the new contribution SOM reside within the **M** (Monitoring) mode. 110

5.2 SAARM T-Matrix for $i = 1...I$ All-Source Sensors [44]. If Sensor 3 were faulty, for example, each subfilter that includes that sensor would report a fault (all across row three) and only subfilter 3 would remain consistent. Filter 3 would then be promoted to the main filter level and a new bank of FDE subfilters must be populated. 114

5.3 sUAS Flight Test Data, 12 Oct 2018, Camp Atterbury, IN 116

5.4 Trace of *a posteriori* Layer 1 Subfilter Position Covariance, Simulated Single GPS 15 Pseudorange Sensor Failure 117

5.5 Trace of *a posteriori* Layer 2 Subfilter Position Covariance, Simulated Single GPS 15 Pseudorange Sensor Failure 118

5.9 Scenario 1 State Estimation Error for 1 Run with Sensor Anomaly from t = 240 to 330 sec and Biased Sensor added at t = 360 sec 132

5.10 Scenario 1 Mean 3-D RSS Error ± 1-σ for 1000 Runs with Sensor Anomaly from t = 240 to 330 sec and Biased Sensor added at t = 360 sec 133

5.11 Scenario 2 State Estimation Error for 1 Run with Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec 135

Figure Page

5.12 Scenario 2 Mean 3-D RSS Error ± 1-σ for 1000 Runs with Sensor Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec 136

5.13 Scenario 3 State Estimation Error for 1 Run with Sensor Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec . . 138

5.14 Scenario 3 Mean 3-D RSS Error ± 1-σ for 1000 Runs with Sensor Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec 139

5.15 Scenario 4 State Estimation Error for 1 Run with Sensor Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec . . 141

5.16 Scenario 4 Mean 3-D RSS Error ± 1-σ for 1000 Runs with Sensor Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec 142

5.6 ARMAS Framework with Novel Observability Layer for Resiliency to One Simultaneous Sensor Failure. 144

5.7 Sample of 10-SV Stationary Constellation Skyplot with Discrete Random Uniform Elevations and 1 Satellite Directly Overhead 145

5.8 Sample Truth Trajectories, 300 Runs 146

5.17 Summary of Detection Rates for 40 meter Biased Pseudorange Sensor 147

6.1 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. 149

6.2 ARMAS Framework State Diagram [43] 157

Figure Page

6.3 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) 175

6.4 Stingy Collaboration Example: Wingman 1 receives corrupted collaborative donor information from Wingman 3. A single fault detection and exclusion action is required to remove all historical contributions from Wingman 3. . . . 176

6.5 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. 177

6.6 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) 178

6.7 Combined Overhead and Profile View of WNG1 Split Collaborative "GPZ-C" and Proprioceptive "GPZ-P" ARMAS-SOM Instances with individual Layer 1 Subfilter Solutions at T = 13m 19s (T = 799s) inside the L1 Jamming + L2 Spoofing Region 3 179

6.8 Size Comparison of Split "Stingy" Collaborative and Proprioceptive All-source Guaranteed Position Zones for WNG1 in GNSS Regions 1-4 180

6.9 Estimator Credibility Comparison for Split Collaborative and Proprioceptive Instances of ARMAS-SOM for 5 Vehicles 181

List of Tables

Table Page

2.1 Position MSE for Distributed Localization Scenario [m] 43

2.2 Average NEES for Distributed Localization Scenario 44

3.1 Mean Position MSE for 10K Trials, T_{trial} = 150 sec, σ_{range} = 5 m 65

4.1 Grand Mean (Median) 2-D Position RMSE for 10-Vehicle Network with Varying Mission Vehicle Weights, (4) 10K Monte Carlo Trials 96

5.1 Scenario 1 Results: RSS Error, GPS4 Exclusion Magnitude, Detection Rate, and Quantity of Augmented Sensors 131

5.2 Scenario 2 Results: RSS Error, GPS4 Exclusion Magnitude, Detection Rate, and Quantity of Augmented Sensors 134

5.3 Scenario 3 Results: RSS Error, GPS4 Exclusion Magnitude, Detection Rate, and Quantity of Augmented Sensors 137

5.4 Scenario 4 Results: RSS Error, GPS4 Exclusion Magnitude, Detection Rate, and Quantity of Augmented Sensors 140

6.1 Comparison of Proprioceptive and Collaborative ARMAS-SOM Instances . . . 174

List of Acronyms

Acronym	Definition
ASPN	All Source Positioning and Navigation
ARMAS	Autonomous and Resilient Management of All-source Sensors
CISA	Cyber and Infrastructure Security Agency
DHS	Department of Homeland Security
EKF	Extended Kalman Filter
FDE	Fault Detection and Exclusion
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GPZ	Guaranteed Position Zone
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
KF	Kalman Filter
MEMS	Micro Electro-Mechanical Systems
NAVWAR	Navigation Warfare
PNT	Positioning, Navigation, and Timing
RAIM	Receiver Autonomous Integrity Monitoring
RSR	Resilient Sensor Recovery
SAS	Self-Aligning Swarm
SCANS	Swarm Control for Autonomous Navigation Support
SAARM	Sensor-Agnostic All-source Residual Monitoring
SOM	Stable Observability Monitoring
sUAS	small Unmanned Airborne System

AN ENHANCED FRAMEWORK FOR COLLABORATIVE ALL-SOURCE NAVIGATION WITH INTEGRITY

I. Introduction

With Global Navigation Satellite System (GNSS) navigation, simultaneously redundant, synchronous measurements with validated measurement models are readily available. In GNSS degraded and/or denied regions, none of these provisions are guaranteed. All-source navigation fuses a variety of sensor measurements (visual, magnetic, signals of opportunity, etc.) for Positioning, Navigation, and Timing (PNT). Modern challenges require all-source navigation systems to be resilient to various GNSS degradation, denial, and spoofing techniques. In response to these challenges, collaboration allows all-source navigation systems to improve resilience through cooperative information-sharing.

In February 2020, the President of the United States issued Executive Order 13905 "Strengthening National Resilience Through Responsible Use of Positioning, Navigation, and Timing Services" [1]. This "PNT EO" focused on national infrastructure concerns put forth by the Department of Homeland Security (DHS) and the Cyber and Infrastructure Security Agency (CISA). The executive order directed research, development, and pilot testing of additional robust and secure PNT services not dependent on GNSS to support the National PNT Plan. The presented research directly supports the executive order by demonstrating an enhanced framework for collaborative all-source navigation.

The first objective of the 2030 United States Air Force Science and Technology Strategy is to "develop and deliver transformational strategic capabilities" [36]. In this road map, the Air Force expressed a strategic challenge to obtain a mix of capable, trustworthy, autonomous systems that can perform as part of a collaborative system to

provide synergistic, multi-domain effects. With this in mind, this research presents methods for real-time autonomous guidance for single and multi-vehicle cooperative navigation in GNSS-denied environments. The unifying goal of these pursuits is to increase the reliability and trustworthiness of collaborative PNT for critical domestic infrastructure and defense capabilities.

1.1 Summary of Related Research

1.1.1 Vehicle Guidance for Self-Localization.

In GNSS degraded and/or denied regions, vehicles equipped with range or bearing sensors can self-localize via surveillance network multilateration to anchor vehicles, commonly known as "relative navigation". Previous approaches to this problem relied on CPU-intensive global optimization of the Cramer-Rao bound. As Chapter 3 mentions, the computation required to perform this search grows exponentially with search area, making it intractable for real-time trajectory guidance. There are diminishing returns associated with calculating a global optimum for self-localization and a scalable, locally-optimized technique is required.

1.1.2 Cooperative Navigation Support.

Navigation support vehicles can be used to provide persistent autonomous multilateration for nearby vehicles tasked with missions. Given sensor payload, fuel, and cost constraints associated with small Unmanned Airborne System (sUAS), it is desirable to federate costly all-source sensor payloads from mission payloads, especially if the mission vehicles are considered attritable. As Chapter 4 mentions, real-time swarm control requires a scalable processing and communications footprint to enable real-time distributed decision-making. Since navigation support requires swarm vehicles to expend physical resources to maneuver, there is an additional desire to allocate prioritized navigation support for important mission vehicles.

1.1.3 Observability Monitoring.

Autonomous and Resilient Management of All-source Sensors (ARMAS) is an all-source navigation framework [43] which assumes overlapping state observability to detect anomalous sensor behavior and exclude faulty sensors. ARMAS does not provide an intrinsic means to assess this requirement. Unlike GNSS, it is common for all-source sensors to encounter featureless measurement "deserts" which can cause loss of navigation state observability. The ARMAS framework 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 estimates. If anomalous behavior is detected, the ARMAS framework attempts to recover the sensor through recalibration, remodeling, and re-validation. Chapter 5 demonstrates that it is impossible for ARMAS to determine if a sensor is misbehaving or if it can be re-validated if overlapping state observability is lost. A method to establish intrinsic information awareness is required to provide a warning when fundamental observability assumptions are in danger. This warning should be used to provide a timely indication to perform collaborative augmentation before fundamental framework assumptions are violated.

1.1.4 Collaborative Navigation.

Collaborative navigation can result in dependency between state estimates and measurements. If a relative measurement is later re-used without taking historical dependencies into account, this can result in double-counting. Distributed networks are scalable and resilient to changes in topology but are particularly vulnerable to this problem. If dependencies are not properly managed, the estimator can become inconsistent by overestimating available sensor information and eventually diverge. Since estimator corruption is always discovered after the fault occurs, it is important to employ a scalable

architecture in a manner which facilitates the recovery of consistency after collaborative fusion occurs. As mentioned in Chapter 6, existing collaborative navigation techniques either prevent the recovery of previously fused information or are impractical due to lack of scalability resulting from processing and communications overhead. A scalable framework which can correctly fuse collaborative information and recover estimator consistency post-fault is required.

1.2 Research Contributions

We have defined the research problem of real-time autonomous vehicle guidance for both self-localization and cooperative offboard navigation support. Several gaps in this field have been identified which are addressed through demonstrations of scalable distributed control schemes. We have identified a critical intrinsic observability monitoring feature missing from the ARMAS framework. This research addresses this problem with a novel navigation state observability monitoring technique which seamlessly integrates with ARMAS and is able to assess current vulnerability in the event of an unknown sensor failure. We have defined the collaborative all-source navigation problem and identified specific gaps in this field of research. This dissertation addresses these gaps with an enhanced 3-D demonstration of a novel selective "stingy" collaborative framework which provides intrinsic awareness to augment with offboard information, maintains consistent resilient all-source navigation, and preserves the ability to gracefully recover autonomy.

The following list enumerates the contributions of this research (categorized by topic):

1. **Single-Vehicle Trajectory Optimization for Self-Localization**

   I Introduces a novel locally-optimized technique with a scalable footprint for real-time vehicle guidance.

II Demonstrates that self-localization RMSE is reduced by 17% when compared to formation flight.

III Demonstrates the optimized error distribution approaches Normality (Non-optimized position RMSE is highly skewed).

2. **Swarm Control for Autonomous Navigation Support**

I Proves distributed local optimization of a globally convex cost surface can result in consensus-like behavior.

II Demonstrates novel real-time swarm guidance for weighted offboard navigation support.

III Demonstrates a novel scalable, real-time swarm guidance technique in large swarms with 50 or more agents.

IV Demonstrates a 36% reduction in supported vehicle position RMSE for optimized navigation support versus non-supported formation flight.

V Demonstrates prioritized mission vehicle position RMSE distributions approach Normality (non-supported vehicle position RMSE is highly skewed).

3. **Stable Observability Monitoring for Intrinsic Navigation State Awareness**

I Provides a novel 3-D position state observability margin, a critical missing component of ARMAS.

II Demonstrates a novel method to detect when overlapping state observability is threatened.

III Demonstrates seamless ARMAS integration in a 3-D GNSS simulation environment.

IV Provides a timely warning to augment ARMAS with additional sensor data.

V Demonstrates reduced estimation error, preservation of fault detection and exclusion consistency, and timely intrinsic information awareness versus to legacy ARMAS.

4. **A Resilient Collaborative All-source Navigation Framework with Integrity**

I Demonstrates a novel resilient collaborative 3-D all-source navigation framework.

II Demonstrates resilience to inconsistency resulting from onboard corruption, offboard corruption, or cross-correlation.

III Demonstrates a novel scalable "stingy collaboration" architecture to fuse collaborative information in a manner which facilitates the recovery of consistency after fault detection.

IV Preserves the ability to gracefully recover autonomy and understand when it is safe to do so.

V Demonstrates reduced estimation error, increased navigation resilience, and preservation of all-source integrity when compared to legacy ARMAS in a realistic 3-D GNSS denial/spoofing scenario.

1.3 Outline

This dissertation is organized into six additional chapters. Chapter 2 contains mathematical notation, background, and a brief history of all-source navigation required to describe the collaborative all-source navigation problem. The ARMAS navigation framework [43] and Sensor-Agnostic All-source Residual Monitoring (SAARM) algorithm [44] are summarized. Next, an overview of stochastic observability methods is provided in context of navigation position state analysis. This section concludes with a brief summary of current cross-correlation management techniques and estimator credibility analysis methods which are used to support fusion of off-board collaborative sensor information.

Chapters 3 through 6 contain background, simulation methodology, results, and conclusions for the individual subset of topics contained in each. Chapter 3 contributes a real-time heuristic method of trajectory generation Self-Aligning Swarm (SAS) for a single aircraft to minimize relative positioning error in a wireless sensor network composed of a simulated formation of maneuvering aircraft. This control scheme, SAS, is based on a compact, scalable particle swarm to locally optimize Fisher Information in real-time. Chapter 4 presents a distributed real-time autonomous navigation control scheme with the introduction of the Swarm Control for Autonomous Navigation Support (SCANS) algorithm. SCANS is a real-time distributed swarm control scheme for autonomous navigation support vehicles operating in a distributed surveillance network to optimize multilateration for a set of non-cooperative maneuvering offboard mission vehicles. This effort builds on the concept of local optimization of globally convex cost surfaces (introduced in Chapter 3) to demonstrate prioritized consensus-like multilateration optimization for non-cooperative mission aircraft in a surveillance network.

Chapter 5 contributes the definition and assessment of a position state observability margin which provides intrinsic navigation state awareness. If this margin is threatened, a warning can be issued to proactively augment navigation state observability with collaborative support from nearby members in a distributed wireless network. The intrinsic navigation state awareness indicates when collaborative information links are no longer required to maintain resilience. Chapter 6 introduces a framework which builds on the fault detection and exclusion, online calibration, and remodeling of ARMAS using observability monitoring described in Chapter 5. A novel "stingy" cross-correlation management method is introduced to provide scalable resilient collaborative all-source navigation with the option to return to autonomy. Chapter 7 summarizes the entire research effort, major contributions, and provides guidelines to consider when constructing a resilient collaborative navigation framework.

II. Background

This chapter summarizes relevant methods and concepts required to tackle the collaborative all-source navigation problem. The chapter begins by establishing notation conventions and reference frames used throughout our research. Next, summaries of recursive Bayesian estimation, residual monitoring, inertial navigation, and all-source navigation are provided to motivate our proposed efforts. Next, the resilient all-source ARMAS navigation framework and SAARM algorithms, which underpin this effort, are discussed. An overview of stochastic observability methods is discussed in the context of navigation position state analysis. Next, we analyze several current cross-correlation management techniques which can be used to support fusion of off-board collaborative sensor information.

2.1 Notation Conventions

Scalars: Scalars are represented with lower and uppercase characters in *italics*, e.g., A or a.

Vectors: Vectors are indicated by **bold** lowercase characters e.g., $\mathbf{v}$ or by a single underbar e.g., $\underline{B}$. By default, these are assumed to be column vectors.

Vector Component: The scalar component of a vector is indicated with a subscript, e.g., the i^{th} entry in the vector $\mathbf{v}$ is represented by v_i.

Vector Subscripts: A vector annotated with a subscript (e.g., $\mathbf{v}_i$) indicates the discrete time i sample of the vector $\mathbf{v}$ and is used predominantly within recursive estimation.

A Priori and A Posteriori Estimates: When discussing recursive estimators, like a Kalman filter, it is important to distinguish between a pre-update or "a priori" estimate and a post-update or "a posteriori" estimate, e.g., $\hat{\mathbf{x}}_i^-$ or $\hat{\mathbf{x}}_i^+$.

Matrices: Matrices are indicated by **bold** uppercase characters, e.g. $\mathbf{B}$ or by double underbar e.g., $\underline{\underline{B}}$.

Transpose: The superscript $\mathbf{x}^{\mathrm{T}}$ indicates the transpose of $\mathbf{x}$.

Identity and Zero: The matrix $\mathbf{I}$ is the identity matrix. If not otherwise explained, the dimensions of this matrix will be identified with a subscript.

Estimated Variable: To indicate an estimated quantity or variable, the circumflex or "hat" accent, e.g., $\hat{\mathbf{v}}$ is used.

Statistical Notation: The expected value is represented by $E[x]$. The statistical distribution of a random variable is denoted via $\hookrightarrow$. For example, if a random vector $\mathbf{v}$ at time i follows a Normal distribution with mean $\boldsymbol{\mu}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$, it is represented by $\mathbf{v}_i \hookrightarrow N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$.

Reference Frames: If a vector exists in a certain navigation reference frame, it is indicated with a superscript, e.g., $\mathbf{v}^i$ is a vector in the i-frame.

Direction Cosine Matrices: A DCM indicating a transformation from i to frame e is represented by $\mathbf{C}_i^e$. (e.g. a vector $\mathbf{v}^e$, in the e-frame, may be obtained by the transformation $\mathbf{v}^e = \mathbf{C}_i^e \mathbf{v}^i$).

2.1.1 Reference Frames.

A vehicle's position, velocity, and attitude must be expressed in terms of a reference frame. A comprehensive explanation of navigation reference frames and their utility is provided by [75]. This section briefly summarizes the navigation reference frames required for this research.

True inertial frame (*I*-frame) - a theoretical non-rotating reference frame governed by Newton's laws of motion with no predefined origin or orientation.

Earth-centered inertial frame (*i*-frame) - a non-rotating orthonormal reference frame with origin at the Earth's center of mass. The $x-y$ plane is collocated with the Equatorial plane with the $+x$ axis pointing toward the vernal equinox. The $z-axis$ is aligned with the Earth's poles with the $+z$ axis pointing towards the North Pole. For terrestrial navigation purposes, this is considered an inertial reference frame.

Earth-centered Earth-fixed frame (*e*-frame) - a rotating orthonormal reference frame attached directly to the Earth. The $x-y$ plane is collocated with the Equatorial plane with the $+x$ axis pointing towards the Prime Meridian and $+y$ axis pointing towards 90° East longitude.

Earth-fixed navigation frame (*n*-frame) - an orthonormal reference frame with an origin typically defined near the surface of the Earth, also known as *North-East-Down*. The $+x$ direction points North, the $+y$ direction points East, and the $+z$ direction points Down (direction of acceleration due to Earth's gravity) with respect to the origin. This reference frame is rigidly fixed to the Earth's surface and can be useful for local estimation.

Aircraft body frame (*b*-frame) - an orthonormal reference frame with its origin fixed at the aircraft's center of mass (Fig 2.2). The $+x$, $+y$, and $+z$ axes point out the nose, right wing, and bottom of the aircraft. A strapdown inertial navigation system is typically rigidly attached to the aircraft body frame.

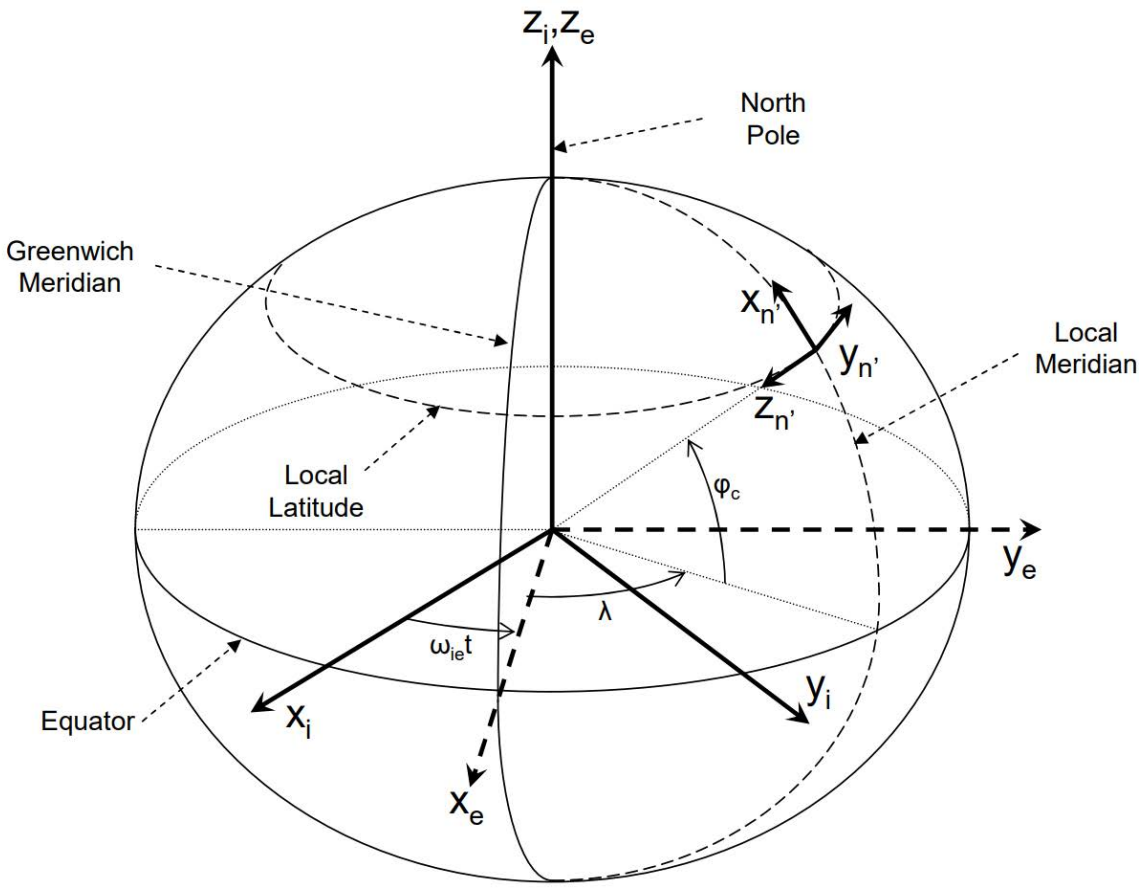

Figure 2.1: Earth-Centered Inertial, Earth-Centered Earth-Fixed, and Earth-Fixed reference frames [24]

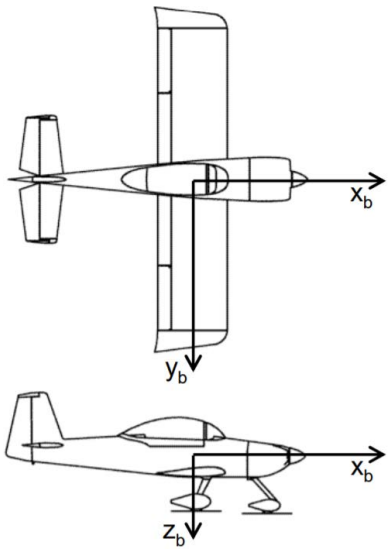

Figure 2.2: Aircraft Body reference frame (*b*-frame) [24]

2.2 The All-Source Navigation Problem

The model estimation notation and reference frame summary compiled in the previous sections provide tools to approach the all-source navigation problem. When applied to airborne vehicles, this discussion often involves computations across multiple reference frames to accurately estimate position, velocity, and attitude in real-time. With GNSS navigation, simultaneously redundant, synchronous measurements with validated measurement models are readily available. In the case of all-source navigation, none of these are guaranteed. The following subsections describe the build-up from classical model estimation to Fault Detection and Exclusion (FDE) based on residual monitoring. This section concludes with a brief evolution of inertial, GNSS, and all-source navigation to provide context for the proposed research.

2.2.1 Recursive Bayesian Estimation.

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 [45], presented

in 1960. 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

$$\dot{\mathbf{x}}(t) = \mathbf{F}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{G}\mathbf{w}(t), \tag{2.1}$$

with

$$E[\mathbf{w}(t)] = 0, \tag{2.2}$$

$$E[\mathbf{w}(t)\mathbf{w}(t)^T(t+\tau] = \mathbf{Q}\delta(\tau), \tag{2.3}$$

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 expectation operator is given by E, time is given by t, $\mathbf{Q}$ is the process noise strength, and $\delta(\tau)$ is the dirac function.

In 1978, the Van Loan method was introduced [76] to provide a method to discretize 2.1 and 2.3 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, Δt.

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

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \tag{2.4}$$

where

$$E[\mathbf{v}_k] = 0, \tag{2.5}$$

$$E[\mathbf{v}_k\mathbf{v}_{k+\Delta t}^T] = \mathbf{R}\delta_{k,k+\Delta t}, \tag{2.6}$$

$\mathbf{z}$ is the sensor measurement vector, k is the discrete sample index, and $\mathbf{v}$ is the discrete-time measurement noise instance.

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

$$\hat{\mathbf{x}}_{\mathbf{k+1}}^{-} = \mathbf{\Phi}\hat{\mathbf{x}}_{\mathbf{k}}^{+} + \mathbf{B_d}\mathbf{u_k}, \tag{2.7}$$

$$\mathbf{P}_{\mathbf{k+1}}^{-} = \mathbf{\Phi}\mathbf{P}_{\mathbf{k}}^{+}\mathbf{\Phi}^{\mathbf{T}} + \mathbf{Q_d}. \tag{2.8}$$

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

$$\mathbf{K_k} = \mathbf{P}_{\mathbf{k}}^{-}\mathbf{H}^{\mathbf{T}}[\mathbf{H}\mathbf{P_k}\mathbf{H}^{T} + \mathbf{R}]^{-1}, \tag{2.9}$$

$$\hat{\mathbf{x}}_k^{+} = \hat{\mathbf{x}}_k^{-} + \mathbf{K}_k[\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^{-}], \tag{2.10}$$

$$\mathbf{P}_k^{+} = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\mathbf{P}_k^{-}. \tag{2.11}$$

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.

2.2.2 Basic 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 [56]. 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 SAARM algorithm implemented by the ARMAS framework.

After a sensor measurement $\mathbf{z}$ is performed, it is possible to extract a set of KF residuals, $\mathbf{r}$, at time k, resulting in

$$\mathbf{r}_k = \mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^{-}. \tag{2.12}$$

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

$$\mathbf{r}_k \hookrightarrow \mathcal{N}\left(\mathbf{0}, \mathbf{H}\mathbf{P}_k^{-}\mathbf{H}^{\mathrm{T}} + \mathbf{R}\right), \tag{2.13}$$

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

$$p(\mathbf{r}_k|\mathbf{\Sigma}_k) = \frac{1}{\sqrt{|2\pi\mathbf{\Sigma}_k|}} e^{-\frac{1}{2}(\mathbf{r}_k)^{\mathrm{T}}\mathbf{\Sigma}_k^{-1}(\mathbf{r}_k)} \text{ and} \tag{2.14}$$

$$\mathbf{\Sigma}_k = \mathbf{H}\mathbf{P}_k^{-}\mathbf{H}^{\mathrm{T}} + \mathbf{R}. \tag{2.15}$$

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

$$L_{N_k} = \prod_{j=k-N+1}^{k} \frac{1}{\sqrt{|2\pi\mathbf{\Sigma}_j|}} e^{-\frac{1}{2}(\mathbf{r}_j)^{\mathrm{T}}\mathbf{\Sigma}_j^{-1}(\mathbf{r}_j)}. \tag{2.16}$$

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

$$\mathcal{L}_{N_k} = \sum_{j=k-N+1}^{k} -\log(|2\pi\mathbf{\Sigma}_j|) - \frac{1}{2}\mathbf{r}_j^{\mathrm{T}}\mathbf{\Sigma}_j^{-1}\mathbf{r}_j. \tag{2.17}$$

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.

2.2.3 Inertial Navigation.

The Inertial Navigation System (INS) was first developed as a guidance system for early ballistic missiles in the 1940s [68]. Engineers determined that, with a known starting point, it is possible measure and integrate small changes in velocity and rotation to maintain an estimate of a vehicle's location, velocity, and attitude. These measurements

are performed by accelerometers and gyroscopes inside an Inertial Measurement Unit (IMU). The gyroscopes sense rotation rates in the true inertial *I*-frame. Small errors in these measurements accumulate over time to form an integration bias or "drift". These measurement errors constitute the primary error source for inertial navigation. Effects from small rotations of the body frame when traversing over the curvature of the Earth can be measured, resulting in a periodic perturbation known as a Schuler cycle [69]. A computer is often used to keep track of the IMU measurements, resulting in the modern version of INS. The two general categories of INS are strategic-grade "platform" and tactical or Micro Electro-Mechanical Systems (MEMS) grade "strapdown" types [75]. A platform INS is a large, expensive, fully gimbaled unit and is designed to maintain a rigid gyroscopic reference with respect to the direction of Earth's gravity. At any time, a vehicle's current attitude can be directly measured from the platform INS. A strapdown unit is smaller with few (if any) moving parts. Hence its name, the strapdown INS is rigidly mounted to the vehicle and operates in the *b*-frame. The strapdown unit is typically designed with orthogonally-mounted gyroscopes that can measure changes in roll, pitch, and yaw. A computer with a high sample rate is required to measure and integrate changes in vehicle attitude with respect to the *I*-frame and measure specific forces aligned with the principal axes of the *b*-frame. Modern MEMS-grade INS are inexpensive, smaller than 1 cm^3, and are often used to aide sUAS [29] [10].

2.2.4 All-Source Navigation.

As mentioned in the previous section, small cumulative measurement errors form an integration bias or "drift", the primary error source for an INS. Some form of recursive model estimation, like an EKF, is often used to maintain an INS error model and incorporate trusted external geodetic measurements to periodically correct integration bias. In the early 1970s, the Department of Defense initiated a joint program called Global Positioning System (GPS) with a clearly-stated functional baseline which resulted in a constellation of

24 satellites in spatially-separated orbits to maintain assured PNT coverage for an unlimited number of concurrent users [11]. GPS provides a set of trusted absolute geodetic reference measurements and can be used to counter accumulated INS integration bias. Over the next 20 years, GPS became ubiquitous for both military and civilian use. By the 1990s, accurate and available GPS was a common assumption. Civilian GPS accuracy was improved with the removal of selective availability in 2000 [11]. Over this period, there was significant motivation for coupled GPS/INS navigation research [16][72] [64][28]. Consequently, efforts in non-GPS navigation methods were often relegated into the alternative navigation category, mainly suitable for environments where GPS was unavailable such as underwater or indoors [66] [54]. Around the same time, localized GPS denial via noise jammers began to proliferate. Additionally, the possibility of advanced GPS emulation or "spoofing" was recognized [17][61][58]. In response, NATO coined the term Navigation Warfare (NAVWAR) as a method to prevent the hostile use of PNT information while protecting unimpeded use of this information for allied forces. Under this paradigm, GPS signals can no longer be assumed as an available or trusted source [61]. In 2010, DARPA responded to this with the announcement of the All Source Positioning and Navigation (ASPN) program [74]. The ANT Center at AFIT embraced all-source navigation as a means to address NAVWAR by combining all available sensor information for PNT [23]. Since 2005, there has been significant research in alternative navigation technologies such as Visual [77], Signals of Opportunity [20], Magnetic [13], and others [23][63][22]. In 2018, ARMAS [43] was introduced to provide a generalized framework for resilient management of all-source sensors which leveraged SCORPION pluggable Bayesian estimators to gracefully manage multiple concurrent state estimation subfilters [46]. In 2019, the monitoring mode of the ARMAS framework was redesigned to provide a means for all-source integrity, known as SAARM [44]. The following sections summarize the contributions of both ARMAS and SAARM and highlight motivations for collaborative all-source navigation with integrity.

2.3 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 [43]. 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 to address the following nonlinear navigation problem:

$$\dot{\mathbf{x}}(t) = \boldsymbol{f}[\mathbf{x}(t), \boldsymbol{\epsilon}(t), \boldsymbol{u}(t), t] + \boldsymbol{G}(t)\boldsymbol{w}(t) \tag{2.18}$$

where $\mathbf{x}$ is a $N \times 1$ state vector of a vehicle's position, velocity, and attitude. The measurement error states vector $\boldsymbol{\epsilon}$ 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:

$$\mathbf{z}_k^{[j]} = \mathbf{h}^{[j]}[\mathbf{x}(t), \boldsymbol{\epsilon}^{[j]}(t), \mathbf{u}(t), t, \mathbf{p}^{[j]}] + \boldsymbol{v}_k^{[j]}, \tag{2.19}$$

where k is the discrete time index, t is continuous time, $\mathbf{h}^{[j]}$ is the nonlinear measurement function for the j^{th} sensor, $\boldsymbol{\epsilon}^{[j]}$ is an $L \times 1$ subset of $\boldsymbol{\epsilon}$ 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 resulting from the measurement noise $\mathbf{Q}$ 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

$$\mathbf{r}_k^{[j]} = \mathbf{z}_k^{[j]} - \mathbf{h}^{[j]}\left[\hat{\mathbf{x}}_k^-, \hat{\boldsymbol{\epsilon}}_k^{[j]-}, \mathbf{u}_k, t_k, \hat{\mathbf{p}}_k^{[j]}\right], \tag{2.20}$$

where $\hat{\mathbf{x}}_k^-$, $\hat{\boldsymbol{\epsilon}}_k^{[j]-}$, and $\hat{\mathbf{p}}_k^{[j]}$ are estimated quantities. Assuming white Gaussian noise, the measurement residual from 2.20 is expected to follow the distribution

$$\mathbf{r}_k^{[j]} \hookrightarrow \mathcal{N}\left(\mathbf{0}_{N\times 1}, \mathbf{S}_k^{[j]}\right), \tag{2.21}$$

$$\mathbf{S}_k^{[j]} = \mathbf{H}_k^{[j]}\mathbf{P}_k^-\mathbf{H}_k^{[j]^T} + \mathbf{R}_k^{[j]}, \tag{2.22}$$

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 matrix 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 (2.17) to monitor the statistical distribution of a user-defined monitoring period composed of recent Kalman pre-update residuals. A Chi-square, χ^*, test statistic is used to detect excursions outside a user-defined threshold across the sampling period. Sensors in validation mode are excluded from impacting the main state estimates. Trusted sensors are directly 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 2.3.1.

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 $\boldsymbol{\epsilon}^{[j]}$ 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 $\boldsymbol{\epsilon}^{[j]}$), 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 (2.21) 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 2.3 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.

One assumption that is not explicitly discussed in [43] is that ARMAS requires overlapping state observability [31] to detect anomalous sensor behavior. 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 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.

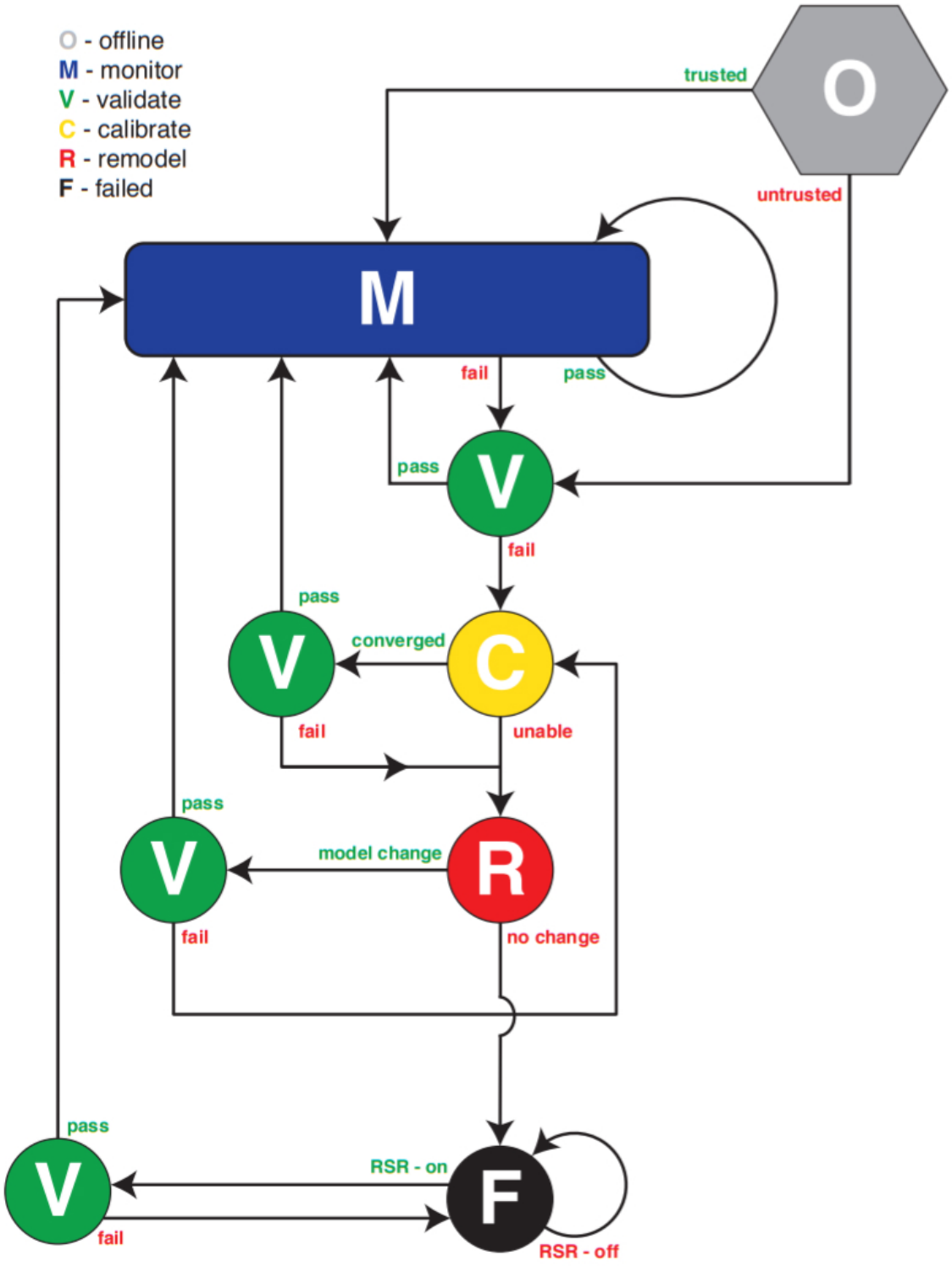

Figure 2.3: ARMAS Framework State Diagram [43]

2.3.1 Sensor-Agnostic All-source Residual Monitoring (SAARM) .

As previously alluded, the monitoring mode in ARMAS was redesigned in 2019 with 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 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

$$\dot{\mathbf{x}}(t) = \boldsymbol{f}[\mathbf{x}(t), \boldsymbol{\epsilon}(t), \boldsymbol{u}(t), t] + \boldsymbol{G}(t)\boldsymbol{w}(t) \tag{2.23}$$

where $\mathbf{x}$ is a $N \times 1$ state vector of a vehicle's position, velocity, and attitude. The measurement error states vector $\boldsymbol{\epsilon}$ 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

$$\mathbf{z}^{[i]}(t_k) = \mathbf{h}^{[i]}[\hat{\mathbf{x}}^{[j]}(t_k^-), \mathbf{u}(t_k), t_k] \tag{2.24}$$

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

$$\hat{\mathbf{z}}^{[i,j]}(t_k) = \mathbf{h}^{[i]}[\hat{\mathbf{x}}^{[j]}(t_k^-), \mathbf{u}(t_k), t_k], \tag{2.25}$$

where the estimated covariance matrix is defined by

$$\mathbf{P}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}{}^{[i,j]}(t_k^-) = \mathbf{H}^{[i]}(t_k^-)\mathbf{P}_{\hat{\mathbf{x}}\hat{\mathbf{x}}}{}^{[j]}(t_k^-)\mathbf{H}^{[i]}(t_k^-)^T. \tag{2.26}$$

Using (2.25) and (2.26), the "pre-update residual" vector between sensor i and filter j, $\mathbf{r}^{[i,j]}$ and its covariance matrix, $\mathbf{P}_{rr}^{[i,j]}$ are defined as

$$\mathbf{r}^{[i,j]}(t_k) = \mathbf{z}^{[i]}(t_k) - \hat{\mathbf{z}}^{[i,j]}(t_k^-), \tag{2.27}$$

$$\mathbf{P}_{rr}^{[i,j]}(t_k) = \mathbf{R}^{[i]}(t_k) + \mathbf{P}_{\hat{z}\hat{z}}^{[i,j]}(t_k^-). \tag{2.28}$$

Fault detection relies on computing a moving average of recent residual-space test statistics formed by pre-update residual vectors from (2.27) and (2.28). 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

$$d^2 = (\mathbf{y} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{y} - \boldsymbol{\mu}), \tag{2.29}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean and covariance of a Z_i-dimensional Gaussian distribution. It is known that a sum of M independent d^2 distances follows a χ^* distribution with M degrees of freedom [21] given by

$$\chi^* = \sum_{s=k}^{k+M} d^2(t_s), \tag{2.30}$$

$$d^2(t_k) = \mathbf{r}^T(t_k)[\mathbf{P}_{rr}(t_k)]^{-1}\mathbf{r}(t_k). \tag{2.31}$$

The set of pre-update residuals is known to be a zero-mean, white sequence [56]. The fault detection test for M pre-residuals is composed of the following hypotheses:

$$H_0 : \chi^*_{[i,j]} < \chi^2(1 - \alpha/2, M \times Z_i) \tag{2.32}$$

$$H_1 : \chi^*_{[i,j]} > \chi^2(1 - \alpha/2, M \times Z_i) \tag{2.33}$$

where α is the probablity of false alarm and M is the number of averaged pre-residual samples. 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). 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* [43].

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

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

Figure 2.4 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 2.3. 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:

$$J_N = \binom{I}{I-N} = \frac{I!}{N!(I-N)!}. \tag{2.34}$$

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 SAARM subfilters!

As a result of the uncorrupted subfilter guarantee provided by the ARMAS framework for a single simultaneous fault, 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* [44]. 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. Figure 2.5 shows a visualization of SAARM all-source horizontal position integrity

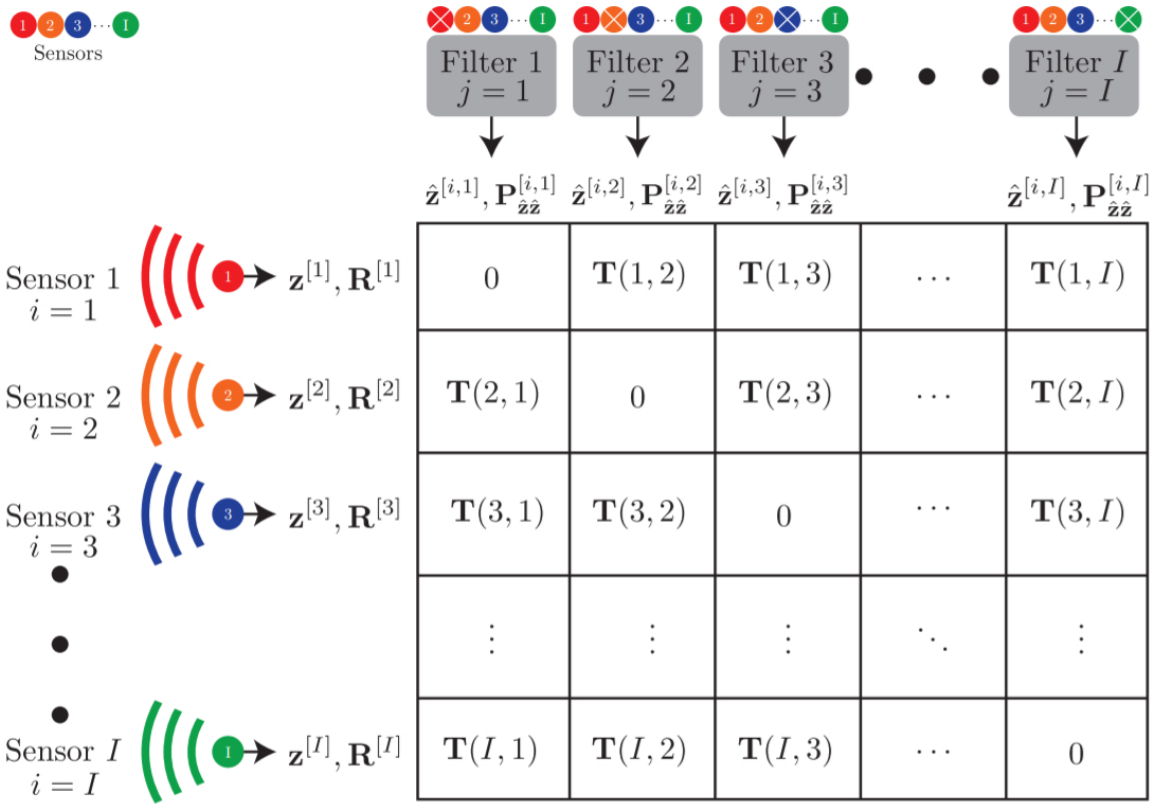

Figure 2.4: SAARM T-Matrix for $i = 1...I$ All-Source Sensors [44]

recovery after a single sensor fault. The union of position error covariance ellipses for the remaining $J - 1 = 4$ subfilters is shown with a green border.

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 χ^* statistical tests of pre-update measurement residuals. Fault exclusion is based on a subfilter consensus approach which is tallied in a novel $\mathbf{T}$ matrix. A technique is introduced to provide resiliency to multiple simultaneous faults at the expense of factorial growth in the required quantity of concurrent estimation filters. 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.

2.4 Observability Assessment Methods

The previous section summarized the SAARM algorithm used to perform fault detection, exclusion, and integrity functions within the ARMAS framework. 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. To ensure this assumption is not violated, we believe the designer should continuously monitor overlapping observability in every ARMAS subfilter. The following sections summarize general eigenvalue decomposition and normalization for the error covariance matrix, $\mathbf{P}$, and a subsequent degree of observability method designed to numerically assess specific state observability within a nonlinear system.

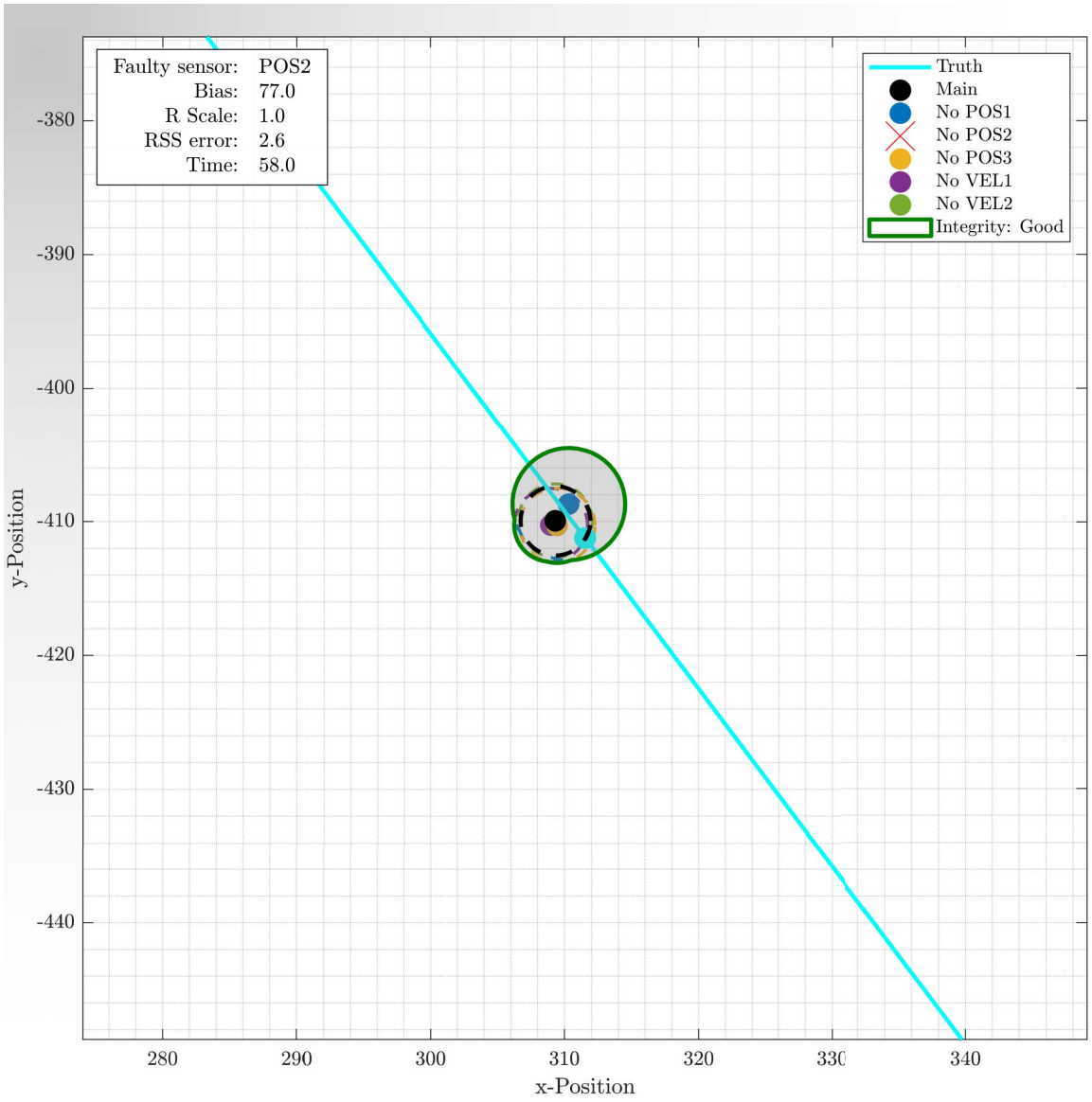

Figure 2.5: SAARM All-Source Horizontal Position Integrity (SCORPION Pluggable EKF)

2.4.1 Eigenvalue Normalization for the Error Covariance Matrix.

In 1983, a general method for eigenvalue normalization of the state estimate error covariance matrix, $\mathbf{P}$, was introduced [31]. This method set a bound on the eigenvalues and forced them to be dimensionless. Both of these attributes facilitate interpretation of state estimate observability. The pre-update error covariance matrix, $\mathbf{P}^{-}(t_k)$, is assumed to be positive definite. The post update error covariance, $\mathbf{P}^{+}(t_k)$ is normalized using the congruent transformation (2.35):

$$\mathbf{P}^{+}_{norm}(t_k) = \left(\sqrt{\mathbf{P}^{-}(t_k)}^{-1} \mathbf{P}^{+}(t_k) \sqrt{\mathbf{P}^{-}(t_k)}^{-1} \right) \tag{2.35}$$

where $\sqrt{\mathbf{P}}$ is the element-wise square root of the diagonal terms of the symmetric matrix $\mathbf{P}$. The largest eigenvalue of $\mathbf{P}^{+}(t_k)$ indicates the variance of the state(s) that are least observable and the rest of the eigenvalues are normalized based on this value. Expectedly, the states with the smallest eigenvalues indicate higher observability. The sum of the eigenvalues in the normalized post update error covariance matrix, $\mathbf{P}^{+}_{norm}(t_k)$, are forced to be equal to $trace\left(\mathbf{P}^{+}_{norm}(t_k)\right)$. This is performed so the eigenvalues could easily be scaled to the order of the system, n, via (2.36)

$$\mathbf{P}^{+}_{norm}(t_k) = \frac{n}{trace(\mathbf{P}^{+}(t_k))} \mathbf{P}^{+}(t_k). \tag{2.36}$$

This basic normalization method provides a convenient way to assess the degree of observability across all system states. Since all-source sensors are likely heterogeneous and/or asynchronous, this observability information is particularly important to assess FDE and integrity performance within the ARMAS framework. The next subsection summarizes a slightly more advanced method, which quantifies observability for a specific set of states in a nonlinear system.

2.4.2 Degree of Observability.

In 2017, a method to quantify the observability of specific state estimates in nonlinear systems was developed [60]. Given a discrete, nonlinear system of the form

$$\mathbf{x}_{k+1} = \mathbf{\Phi}(t_k, \mathbf{x}_k)\mathbf{x}_k + \mathbf{G}(t_k, \mathbf{x}_x)\mathbf{w}_k, \tag{2.37}$$

$$\mathbf{y}_{k+1} = \mathbf{H}(t_k, \mathbf{x}_k)\mathbf{x}_k + \mathbf{v}_{k+1}, \tag{2.38}$$

the author assumes $\mathbf{w}_k$ and $\mathbf{v}_k$ are uncorrelated and composed of zero-mean, white Gaussian noise components. It is shown that the measurement equation can be rewritten with the observability matrix for this system given by

$$\mathbf{y}_k = \mathbf{O}_k\mathbf{x}_k + \mathbf{v}_k. \tag{2.39}$$

The authors show that if the rank of $\mathbf{O}_k$ is equal to the number of states in the system, n, then every system state is observable. In the fully observable case, the author shows that $\mathbf{O}_k\mathbf{O}_k^T$ is a square invertible matrix. For this case, the following relationship is examined

$$\mathbf{O}_k\mathbf{y}_k = \mathbf{O}_k^T\mathbf{O}_k\mathbf{x}_k + \mathbf{O}_k^T\mathbf{v}_k.$$

Since $\mathbf{O}_k$ is invertible, it is possible to solve for the state estimate

$$\mathbf{x}_k = [\mathbf{O}_k^T\mathbf{O}_k]^{-1}\mathbf{O}_k^T(\mathbf{y}_k - \mathbf{v}_k).$$

Once the state estimate is isolated, the matrix pseudoinverse can be recognized and is replaced by the following expression:

$$\zeta_k = \mathbf{O}_k^+\mathbf{y}_k, \tag{2.40}$$

where $\mathbf{O}_k^+ = [\mathbf{O}_k^T\mathbf{O}_k]^{-1}\mathbf{O}_k^T$ for simplification. The scalar form of the i^{th} row of this matrix for $i = 1...n$ system states is described by

$$\zeta_k^i = \alpha_1^i\mathbf{y}_k + \alpha_2^i\mathbf{y}_{k+1} + ... + \alpha_n^i\mathbf{y}_{k+n-1},$$

where α_n^i are in the i^{th} component of the vector ζ_k. Similarly, the measurement equation in scalar form is

$$\upsilon_k^i = \alpha_1^i \mathbf{v}_k + \alpha_2^i \mathbf{v}_{k+1} + ... + \alpha_n^i \mathbf{v}_{k+n-1},$$

where α_n^i are in the i^{th} component of the vector υ_k. The variance of the i^{th} component of measurement noise is

$$\Upsilon_k^i = E\left[(\upsilon_k^i)^2\right] = \left[(\alpha_1^i)^2 + (\alpha_2^i)^2 + ... + (\alpha_n^i)^2\right]\mathbf{r}_k$$

where $\mathbf{r}_k = E[\mathbf{v}_k^2]$, the original system measurement noise variance. Finally, the degree of observability is defined as

$$DOO = \frac{E\left[(\mathbf{x}_k^i)^2\right]\mathbf{r}_k}{E\left[(\zeta_k^i)^2\right]\Upsilon_k^i} = \frac{E\left[(\mathbf{x}_k^i)^2\right]}{E\left[(\zeta_k^i)^2\right]\sum_{j=1}^{n}(\alpha_j^i)^2}. \tag{2.41}$$

Clearly, understanding the measurement noise, measurement model, and the observability matrix are key to assessing the observability of a system state variable. The degree of observability is a scalar and is used to directly quantify the observability of a state-variable without eigenvalue decomposition. The computations required to solve for the degree of observability are relatively straightforward in a Kalman filter algorithm. Since SAARM framework assumes state observability for both FDE and integrity, it is critical that a method for monitoring real-time position state observability is implemented.

2.5 Methods for Managing Cross-Correlated Information

If possible, a vehicle should maintain autonomous navigation using proprioceptive (onboard) sensors. Furthermore, the ARMAS framework requires overlapping state observability. If a vehicle cannot maintain proprioceptive overlapping observability, collaborative navigation support can be used for augmentation. We must be careful because fusing collaborative sensor data 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 [41]. Distributed networks are scalable and resilient to changes in topology but are particularly vulnerable to this problem. If dependencies are not properly managed, the estimator can become inconsistent by overestimating available sensor information and eventually diverge. The following section analyzes several methods to properly account for dependencies when fusing measurements: Covariance Intersection, Conditional Naive Filter, Game-Theoretic Filter, and the Interleaved Update Algorithm. These methods are simulated and assessed based on position MSE and by using a numerical estimator credibility analysis. The Interleaved Update Algorithm is recommended for the proposed research. This section concludes with a method to maintain collaborative resiliency against simultaneous sensor failures.

2.5.1 Covariance Intersection Method.

In [40], Julier proved that the Covariance Intersection (CI) method is guaranteed to avoid the problem of double-counting for any choice of weight w in the interval [0, 1] which guarantees estimator consistency. He admitted the CI method causes a loss in information which results in suboptimal update for loosely correlated measurements.

If noise-corrupted observations are dependent on an estimator's own system state, then independence assumptions are violated. If ignored, these cross-correlations will cause a naive recursive Bayesian estimator to overestimate the amount of information available for an update. A rudementary fix for this problem 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.

The limitations of managing unknown cross-correlation between state estimates and measurements in decentralized estimation with arbitrary network topologies are widely known. It is possible to judiciously adjust Kalman filter tuning to mask the effects of

unknown cross-correlations, but artificially increasing the process noise covariance matrix is likely to cause filter divergence. This motivated the CI method introduced by [41] which fuses the probability densities of two measurements with unknown cross-correlation into a consistent covariance estimate. Per the author's derivation, the two pieces of information, A and B are combined to yield C. The sources are noise corrupted, so they become random variables a, b, and c. The means of vectors $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ are denoted as $\bar{\mathbf{a}}$, $\bar{\mathbf{b}}$ and $\bar{\mathbf{c}}$. The error terms are denoted as $\widetilde{\mathbf{a}} = \mathbf{a} - \bar{\mathbf{a}}$, $\widetilde{\mathbf{b}} = \mathbf{b} - \bar{\mathbf{b}}$, and $\widetilde{\mathbf{c}} = \mathbf{c} - \bar{\mathbf{c}}$. The mean squared error terms (MSE) are

$$\bar{\mathbf{P}}_{\mathbf{aa}} = \mathbf{E}[\widetilde{\mathbf{a}}\widetilde{\mathbf{a}}^{\mathbf{T}}]\ \bar{\mathbf{P}}_{\mathbf{bb}} = \mathbf{E}[\widetilde{\mathbf{b}}\widetilde{\mathbf{b}}^{\mathbf{T}}]\ \bar{\mathbf{P}}_{\mathbf{cc}} = \mathbf{E}[\widetilde{\mathbf{c}}\widetilde{\mathbf{c}}^{\mathbf{T}}]$$

and cross correlation term is

$$\bar{\mathbf{P}}_{\mathbf{ab}} = \mathbf{E}[\mathbf{a}\mathbf{b}^{\mathbf{T}}].$$

The MSE terms are approximated by $\mathbf{P}_{\mathbf{aa}}$ and $\mathbf{P}_{\mathbf{bb}}$. $\bar{\mathbf{P}}_{\mathbf{ab}}$ is assumed to be unknown. Consistency is maintained iff

$$\mathbf{P}_{\mathbf{aa}} - \bar{\mathbf{P}}_{\mathbf{aa}} \geq 0 \text{ and } \mathbf{P}_{\mathbf{bb}} - \bar{\mathbf{P}}_{\mathbf{bb}} \geq 0 \text{ and } \mathbf{P}_{\mathbf{cc}} - \bar{\mathbf{P}}_{\mathbf{cc}} \geq 0,$$

in accordance with the standard definition of consistency [41]. The CI algorithm is described by (2.42) and (2.43)

$$\mathbf{P}_{\mathbf{cc}}^{-\mathbf{1}} = \mathbf{w}\mathbf{P}_{\mathbf{aa}}^{-\mathbf{1}} + (\mathbf{1} - \mathbf{w})\mathbf{P}_{\mathbf{bb}}^{-\mathbf{1}}, \tag{2.42}$$

$$\mathbf{P}_{\mathbf{cc}}^{-\mathbf{1}}\bar{\mathbf{c}} = \mathbf{w}\mathbf{P}_{\mathbf{aa}}^{-\mathbf{1}}\bar{\mathbf{a}} + (\mathbf{1} - \mathbf{w})\mathbf{P}_{\mathbf{bb}}^{-\mathbf{1}}\bar{\mathbf{b}}. \tag{2.43}$$

In the CI algorithm, $w \in [0, 1]$ and blends weighting between parameters $\bar{\mathbf{a}}$ and $\bar{\mathbf{b}}$. w is selected to minimize the $tr(\mathbf{P}_{\mathbf{cc}})$ or $det(\mathbf{P}_{\mathbf{cc}})$, respectively. Minimization of the trace of $\mathbf{P}_{\mathbf{cc}}$ is equivalent to minimizing the primary dimensions of the $\mathbf{P}_{\mathbf{cc}}$ error covariance ellipse. Minimization of the determinant is equivalent to minimization of the area of the $\mathbf{P}_{\mathbf{cc}}$ error covariance ellipse. In [38], Hurley showed the minimization of $det(\mathbf{P}_{\mathbf{cc}})$ is equivalent to

minimization of the Shannon information of the fused probability density function. This insight could be particularly useful if the probability density functions are something other than Gaussian. The CI technique is recommended as a good estimator for highly cross-correlated measurements.

Figures 2.6 and 2.7 show comparisons of the Naive Fusion, CI $det(\mathbf{P_{cc}})$ minimization, and CI $tr(\mathbf{P_{cc}})$ minimization methods for 2 fused measurements, and 4 sequential fused measurements, respectively. It is clear in both cases that NF method provides a much tighter estimate of $\mathbf{P_{cc}}$ than the CI method. This is expected because the NF ignores the cross-correlation terms $\mathbf{P_{ab}}$ as shown in Equations 2.44 and 2.45. This causes the NF to overestimate the available information.

$$\mathbf{P_{cc}^{-1}} = \mathbf{P_{aa}^{-1}} + \mathbf{P_{bb}^{-1}}. \tag{2.44}$$

$$\mathbf{P_{cc}^{-1}}\bar{\mathbf{c}} = \mathbf{P_{aa}^{-1}}\bar{\mathbf{a}} + \mathbf{P_{bb}^{-1}}\bar{\mathbf{b}}. \tag{2.45}$$

The CI method utilizes a conservative estimate of $\mathbf{P_{ab}}$ to guarantee estimator consistency. A downside of the CI method is the likelihood of underestimating available information. The slight differences between the $tr(\mathbf{P_{cc}})$ and $det(\mathbf{P_{cc}})$ CI methods are visible in Figure 2.7 with 4 fused measurements. It should be noted that a highly skewed ellipse will render a small area. For this reason, we chose to implement the $tr(\mathbf{P_{cc}})$ CI method.

2.5.2 Ignore Cross-Correlation Technique (Conditional Naive Filter).

Motivated by potential improvement over the conservative CI method, [2] introduced the idea of ignoring the cross-correlation terms under certain conditions while still maintaining filter consistency. Assuming we have access to consistent estimates of $\widetilde{\mathbf{P}}_{\mathbf{aa}}$ and $\widetilde{\mathbf{P}}_{\mathbf{bb}}$, a consistent estimate of $\widetilde{\mathbf{P}}_{\mathbf{cc}}$ can be obtained from Equation 2.44 under the following conditions for $\bar{\mathbf{P}}_{\mathbf{ab}}$:

$$-\sqrt{\mathbf{P_{aa}}\mathbf{P_{bb}}} \leq \bar{\mathbf{P}}_{\mathbf{ab}} \leq \mathbf{0}, \text{ or} \tag{2.46}$$

$$\left(\frac{\mathbf{P_{aa}^{-1}}\mathbf{P_{bb}^{-1}}}{\mathbf{2}}\right)^{-1} \leq \bar{\mathbf{P}}_{\mathbf{ab}} \leq \sqrt{\mathbf{P_{aa}}\mathbf{P_{bb}}} \tag{2.47}$$

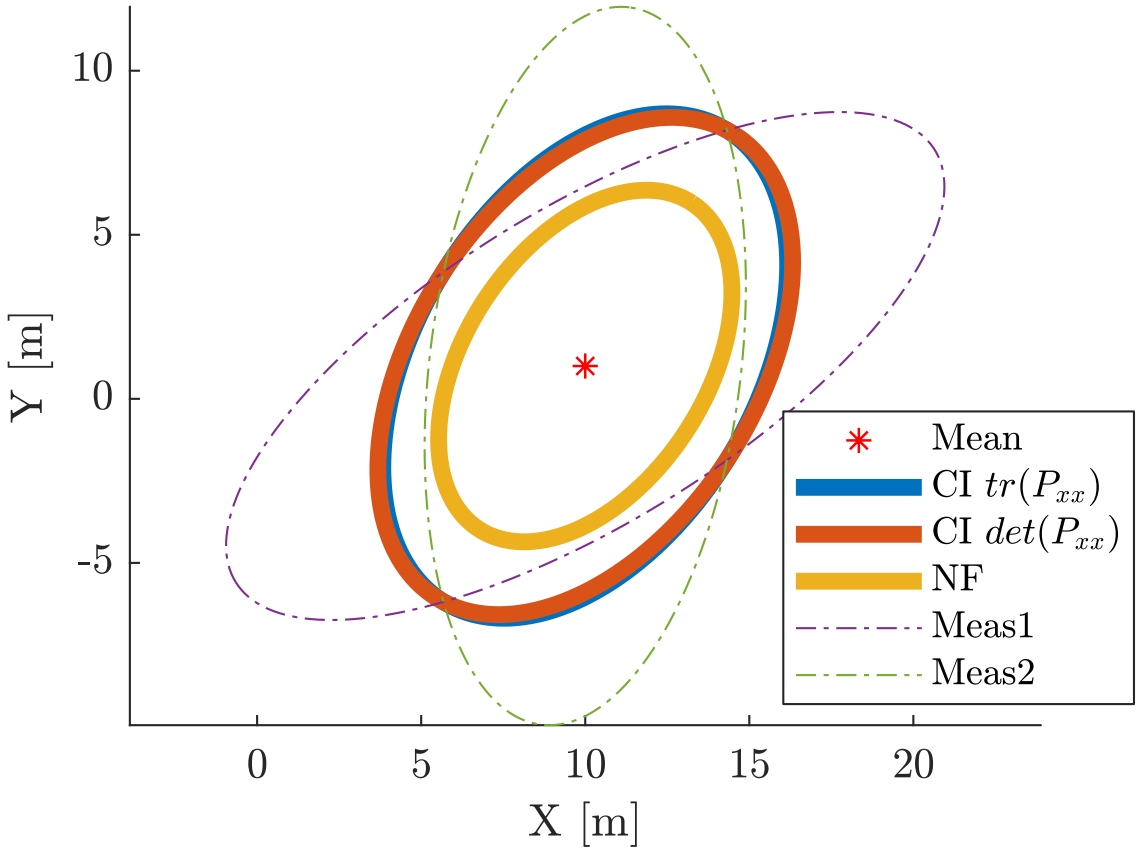

Figure 2.6: Comparison of Fused Error Covariance Ellipses P_{cc} resulting from CI $tr(P_{cc})$ vs. CI $det(P_{xx})$ vs. Naive Kalman Filter Method with 2 Cross-Correlated Range Measurements

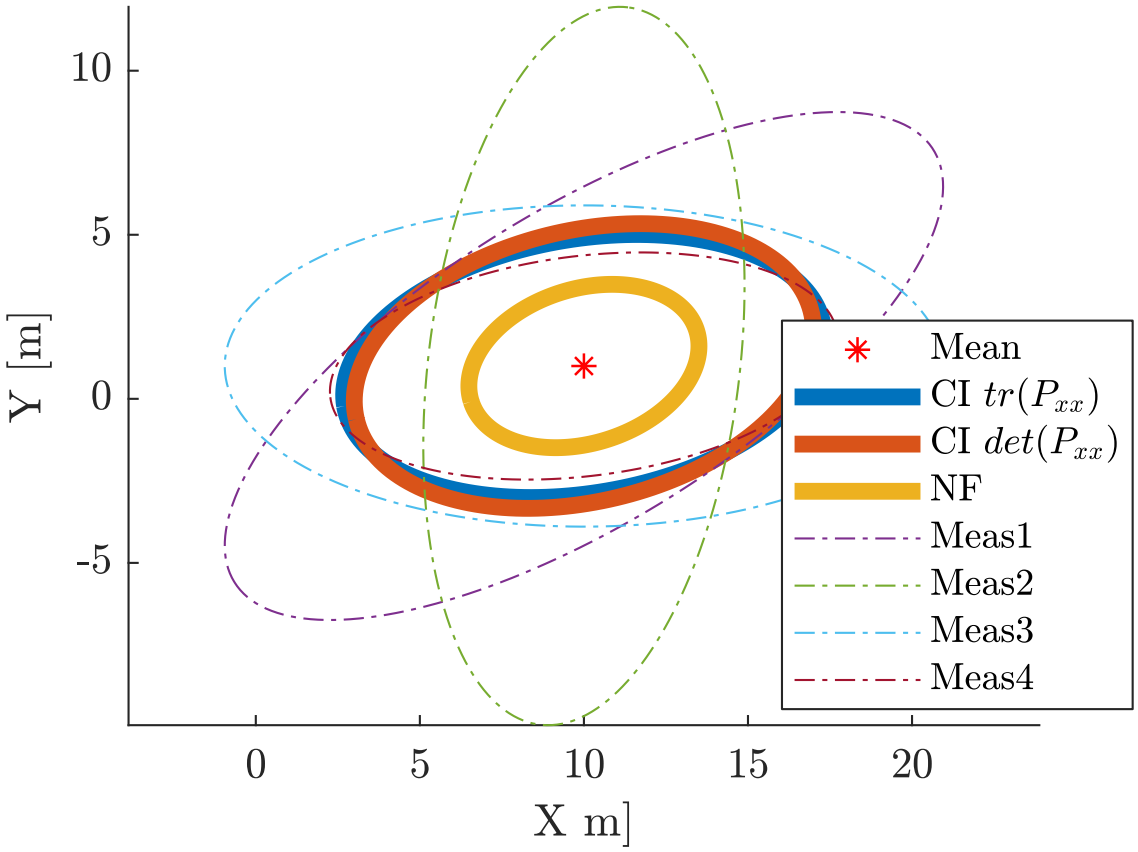

Figure 2.7: Comparison of Fused Error Covariance Ellipses P_{cc} resulting from CI $tr(P_{cc})$ vs. CI $det(P_{xx})$ vs. Naive Kalman Filter Method with 4 Cross-Correlated Range Measurements

Although somewhat insightful, this method requires some knowledge of $\mathbf{P_{ab}}$ to ensure the inequalities defined in Equations 2.46 and 2.47 are met. In other words, one must obtain an accurate estimate of $\bar{\mathbf{P}}_{\mathbf{ab}}$ before deciding between a more conservative method (e.g. CI) or ignoring the cross-correlation terms altogether. Without a technique to accurately estimate cross-correlation, this is not particularly useful.

2.5.3 Game-Theoretic Filter (GTF).

Motivated by fusion accuracy improvement over the CI method, the Game-Theoretic Filter (GTF) [49] provides a novel method to select the weight parameter w from (2.42). In addition to minimizing the $tr(\mathbf{P_{cc}})$ the GTF simultaneously guesses the smallest sufficient value of $\mathbf{P_{ab}}$ in a "convex-concave" game [49]. To perform a measurement update, the GTF simultaneously iterates on estimates for $\mathbf{P_{ab}}$ and w to minimize the $tr(\mathbf{P_{cc}})$ while meeting the following linear matrix inequality constraint:

$$\text{minimize } w \sup_{\mathbf{P_{ab}}} tr(\mathbf{P_{cc}})$$

$$\text{subject to} \begin{bmatrix} \mathbf{P_{aa}} & \mathbf{P_{ab}} \\ \mathbf{P_{ab}}^T & \mathbf{P_{bb}} \end{bmatrix} \geq \mathbf{0}$$

The GTF is designed to select a less conservative estimate of $\mathbf{P_{ab}}$ than the CI method. The GTF seems particularly suited to fusing measurements which contain only a partial state estimate update. However, the GTF may tend to overestimate available information.

2.5.4 Interleaved Update (IU) Algorithm.

The IU algorithm [7] 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 (2.48).

$$\mathbf{P_{i,j}}(t) = \left[\begin{array}{c|c} \mathbf{P_i}(t) & 0 \\ \hline 0 & \mathbf{P_j}(t) \end{array}\right] \tag{2.48}$$

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.

For example, when vehicle 1 first receives an offboard range measurement from nearby vehicle 2 at time t, the current state estimate $\mathbf{x_1}$ and covariance $\mathbf{P_1}$ are copied to form a new set $\mathbf{x_2}$ and $\mathbf{P_2}$ which are updated by the range measurement from vehicle 2. These parameters are separated into a new filter which is propagated and updated separately from $\mathbf{x_1}$ and $\mathbf{P_1}$. A $\mathbf{T_1}(t)$ matrix onboard vehicle 1 is instantiated. Within $\mathbf{T_1}(t)$, row (filter) 2, column (vehicle) 2 is updated with a timestamp for the offboard measurement update from vehicle 2. Equation (2.49) shows how row 1 is updated by vehicle 1's onboard measurements, so (row 1, column 2) of $\mathbf{T_1}(t)$ remains 0.

$$\mathbf{T_1}(t) = \begin{bmatrix} t & 0 \\ t & t \end{bmatrix}. \tag{2.49}$$

At the next time step, no offboard measurements from vehicle 2 are received. As expected, the entry at row (filter) 2, column (vehicle) 2 remains t. Since onboard measurements are received at $t + 1$, the entries in column (vehicle) 1 are updated. Since filter 1 is only updated by vehicle 1, the entry at row (filter) 1, column (vehicle) 2 is always 0. In this case, (2.50) shows the updated $\mathbf{T_i}(t + 1)$ matrix is of the form

$$\mathbf{T_1}(t + 1) = \begin{bmatrix} t + 1 & 0 \\ t + 1 & t \end{bmatrix}. \tag{2.50}$$

The $\mathbf{T_1}(t)$ matrix propagated in (2.49) and (2.50) keeps track of which vehicles have been used to update which filter as well as the age of the last updates.

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 $argmin[tr(\mathbf{P_i^q})]$ 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 [7].

2.5.5 Collaborative Localization Simulation.

We examine three collaborative localization approaches in a simple 2-D scenario with multiple vehicles operating in a distributed wireless sensor network. This simulation was based on an cooperative 4-vehicle scenario [49]. Only one vehicle is equipped with an absolute positioning source (GPS) and all vehicles are equipped with ranging sensors. The sampling rate of every vehicle in the network is fixed at 1 Hz with synchronized measurement updates. The network is monitored by a single centralized Kalman Filter (KF) which can observe all sensor measurements. The GTF, CI, and NF methods operate in a decentralized wireless sensor network with each vehicle operating independently and locally without knowledge of the entire network. Vehicle 1 receives a GPS update on every sample and vehicles 2-4 rely on collaborative localization. All vehicles are able to share their positions over a distributed network and can obtain noise-corrupted ranging to each other. The vehicles are initialized with random Gaussian 2-D velocity with a mean of 0.1 meters per second and maximum velocity of 0.5 meters per second. The vehicles are constrained to a 50 m^2 area. If the vehicles contact the boundary, the velocity component

in the direction of the boundary is inverted so the vehicle "ricochets" to stay within the constrained area.

The range measurements are corrupted by zero-mean independent bivariate Additive White Gaussian Noise (AWGN) from other nearby nodes in 2-D space. In the example, nearby nodes have unique positions separated by true distance D_n and each range observation R_n is corrupted by zero-mean AWGN with magnitude σ_{coop}. Ranging error V_n is known and clock synchronization is assumed.

$$\boldsymbol{R}_n = \boldsymbol{D}_n + \boldsymbol{V}_n, \text{ where } \boldsymbol{V}_n \hookrightarrow \mathcal{N}(\underline{\mu}, \boldsymbol{S}) \tag{2.51}$$

$$\underline{\mu} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \text{ and } \boldsymbol{S} = \sigma^2_{coop} I = \begin{bmatrix} \sigma^2_{coop} & 0 \\ 0 & \sigma^2_{coop} \end{bmatrix} \text{meters}$$

$$\text{with } n = \{1, 2, .., N\} \text{ observations from nearby nodes}$$

$$\text{and } \sigma^2_{coop} = 0.1 \text{ m}^2.$$

At each sample, a node receives a range measurement R_n from each nearby node.

$$\text{with nearby nodes positioned at } \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}, \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}, \ldots \begin{bmatrix} x_n \\ y_n \end{bmatrix}.$$

At each sample, vehicle 1 receives an GPS measurement update with

$$\boldsymbol{R}_{GPS} = \sigma^2_{GPS} I = \begin{bmatrix} \sigma^2_{GPS} & 0 \\ 0 & \sigma^2_{GPS} \end{bmatrix} \text{meters} \tag{2.52}$$

$$\text{and } \sigma^2_{GPS} = 1 \text{ m}^2.$$

Vehicles 2-4 self-localize by combining shared position estimates from nearby vehicles with ranging measurements $\mathbf{R}_N$. These measurements are updated using three distributed estimators (GTF, CI, and NF) and one centralized estimator (KF) for a duration of 200 samples.

2.5.6 Estimator Credibility Analysis.

Estimator performance is always dependent on available measurements. 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 $\widetilde{x}$ of state estimate $\widehat{x}$ at time i is shown by (2.53). The MSE of $\widetilde{x}$ is shown by 2.54.

$$\widetilde{\boldsymbol{x}}_i = \boldsymbol{x}_i - \widehat{\boldsymbol{x}}_i \tag{2.53}$$

$$MSE = \frac{1}{n}\widetilde{\boldsymbol{x}}_i^T\widetilde{\boldsymbol{x}}_i \tag{2.54}$$

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 self-assessed bias and covariance of the state estimate can be used to assess the credibility of an estimator. Li defines estimator credibility as

> "the difference between [an estimator's] actual bias and MSE matrix and its self-assessment (i.e. calculated bias and error covariance) at statistical confidence level α in the sense that the two sets of quantities involved can be treated as equal statistically" [52].

The Normalized Estimation Error Squared (NEES) metric, γ_i, is chi-square distributed, assumes Gaussian errors, and provides a quantitative assessment of the estimator's credibility from pessimistic to optimistic. NEES is calculated according to Eq. 2.55.

$$\gamma_i = (x_i - \widehat{x_i})^T \boldsymbol{P}_i^{-1}(x_i - \widehat{x_i}) \tag{2.55}$$

For an m-dimensional state estimate with n total measurements, the Average NEES is shown by Eq. 2.56.

$$\overline{\gamma} = \frac{1}{nm} \sum_{i=1}^{n} \gamma_i \tag{2.56}$$

NEES represents the square of the distance between the state estimate $\widehat{\boldsymbol{x}}$ and the truth $\boldsymbol{x}$, normalized by the error covariance $\boldsymbol{P}$. Average NEES can be interpreted as a uni-dimensional average of this squared distance. Average NEES quantifies estimator credibility by assessing the accuracy of employed assumptions. 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).

2.5.7 Numerical Results.

Figures 2.8-2.9 show position errors for the Game-Theoretic Filter (GTF), Covariance Intersection (CI) method, Naive Filter (NF), and centralized Kalman Filter (KF) over the 200-sample simulation. The centralized KF optimally combines all available information and is only shown to indicate maximum possible performance. The NF ignores cross-correlation terms and is only shown as a minimum performance threshold. This is expected because ignoring cross-correlation terms causes the filter to overestimate available information which results an overconfident state estimate. The CI and GTF methods generate well-behaved errors. Table 2.1 shows a summary of Position MSE for each method. Analysis indicates the GTF method is able to produce a significant (45%) reduction in Position MSE over the CI method for the collaborative vehicles.

Further analysis of the second moment (covariance) of the state estimate is required to draw further conclusions. Figures 2.10-2.11 show NEES for the 200-sample simulation. The propagated state estimate contains 4 states (x, y position and x, y velocity). For this scenario, (2.55) dictates that a 'perfectly' credible estimator should produce NEES value of 4, equal to the dimensionality of the state estimate.

Both the CI and GTF methods generate nearly identical NEES values for the GPS-equipped vehicle 1. This is expected because the measurement covariance associated with a GPS update is very small compared to a collaborative localization measurement update. Vehicles 2-4 rely on collaborative localization updates and amplify the differences between cross-correlation estimates between the CI and GTF methods. The CI method NEES values never exceed 4. This indicates that the CI method is always pessimistic. In contrast, the GTF makes temporal excursions into optimistic territory as evidenced by values exceeding 4 during the simulation.

Table 2.2 shows a summary of Average NEES for the GTF and CI methods. The optimal value for Average NEES is 1 based on the n term in the denominator of (2.56). Based on the mean distance from 1 for each collaborative vehicle, the GTF method is more credible than the CI method. It is important to note that the GTF method provides an average optimistic assessment for vehicle 2 with an Average NEES value of 1.08. Overly optimistic estimators can be susceptible to divergence. Depending on the application, it may be desired to implement a consistently pessimistic estimator rather than a more credible estimator that is occasionally optimistic.

Table 2.1: Position MSE for Distributed Localization Scenario [m]

Vehicle (Localization Source)	GTF	CI	NF	KF
1 (GPS + Collaborative)	**0.1062**	**0.1100**	**9.6567**	**0.0652**
2 (Collaborative)	**0.6490**	**1.1681**	**10.0114**	**0.0733**
3 (Collaborative)	**0.6599**	**1.1630**	**9.8668**	**0.0761**
4 (Collaborative)	**0.4620**	**0.7779**	**9.7423**	**0.0666**

Table 2.2: Average NEES for Distributed Localization Scenario

Vehicle (Localization Source)	GTF	CI
1 (GPS + Collaborative)	**0.3386**	**0.3472**
2 (Collaborative)	**1.0775**	**0.1848**
3 (Collaborative)	**0.8306**	**0.1703**
4 (Collaborative)	**0.3745**	**0.1784**

2.5.8 Collaborative Resiliency Against Simultaneous Sensor Failures.

The ARMAS framework provides all-source navigation for a single vehicle with seamless transitions between a main EKF and J EKF subfilters corresponding to I all-source sensors. This research intends to modify the ARMAS framework to provide navigation resiliency against simultaneous sensor failures via collaboration. Each tier 1 subfilter is uncorrupted by measurements from one of the all-source sensors. The main filter fuses all available sensor measurements and is primarily used for user output. Assuming I total all source sensors (including offboard sources), ARMAS will maintain a main filter (for user output) and I tier 1 subfilters. To preserve estimator consistency in the event of S simultaneous sensor failures, ARMAS must maintain at least $\binom{I}{S}$ filters. Assuming $I = 7$ sensors, ARMAS must maintain at least $\binom{7}{2} = 21$ tier 2 subfilters to be guaranteed a consistent estimator uncorrupted by measurements from any 2 simultaneous sensor failures.

We have established that the tier S subfilters are each uncorrupted by S sensors. To detect if overlapping obervability is threatened, a stochastic observability test can indicate whether S simultaneous sensor failures would lead to loss of overlapping observability. For example, suppose there are $I = 7$ all-source sensors, 3 of which are off-board

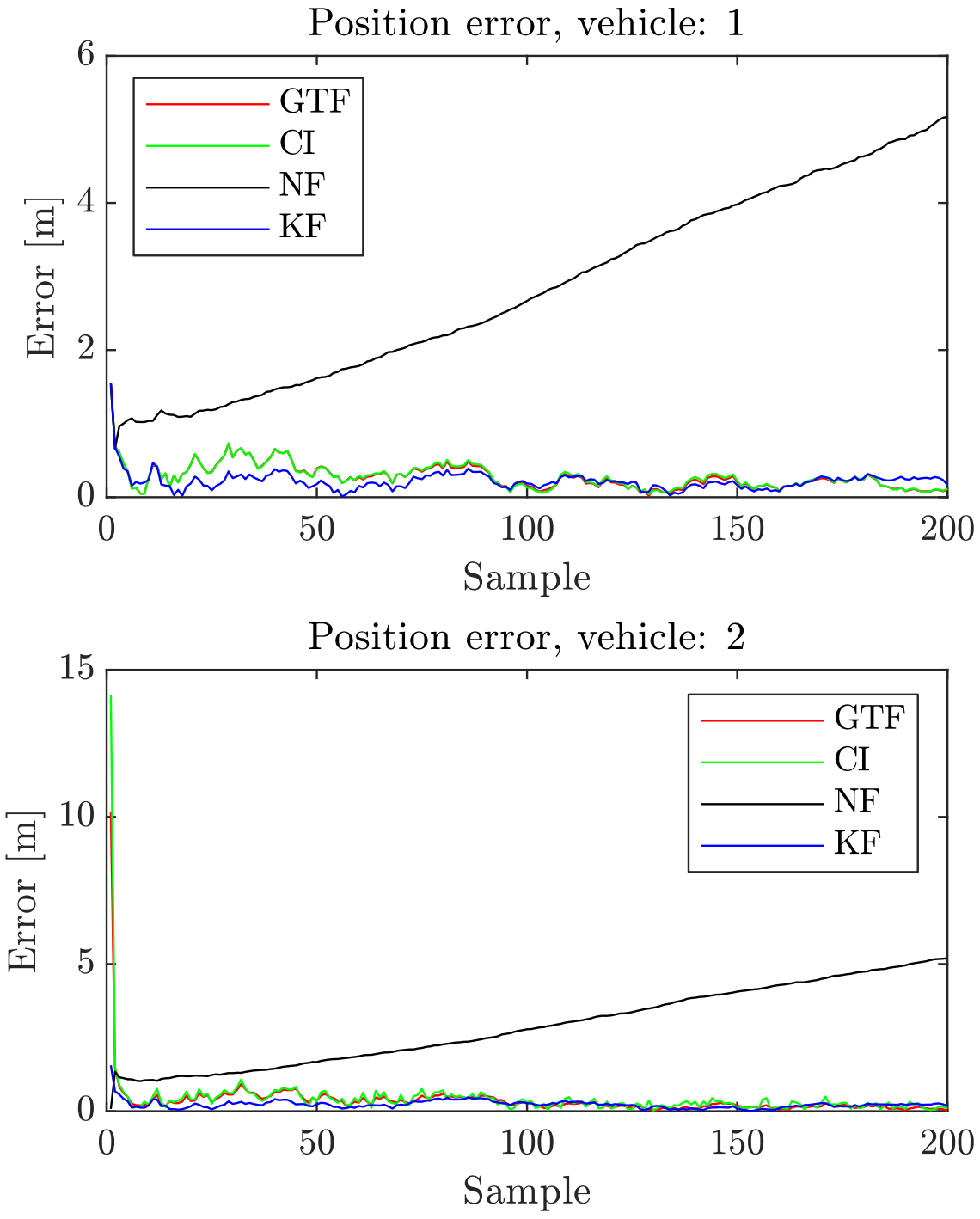

Figure 2.8: Position Error for Collaborative Vehicles 1, 2

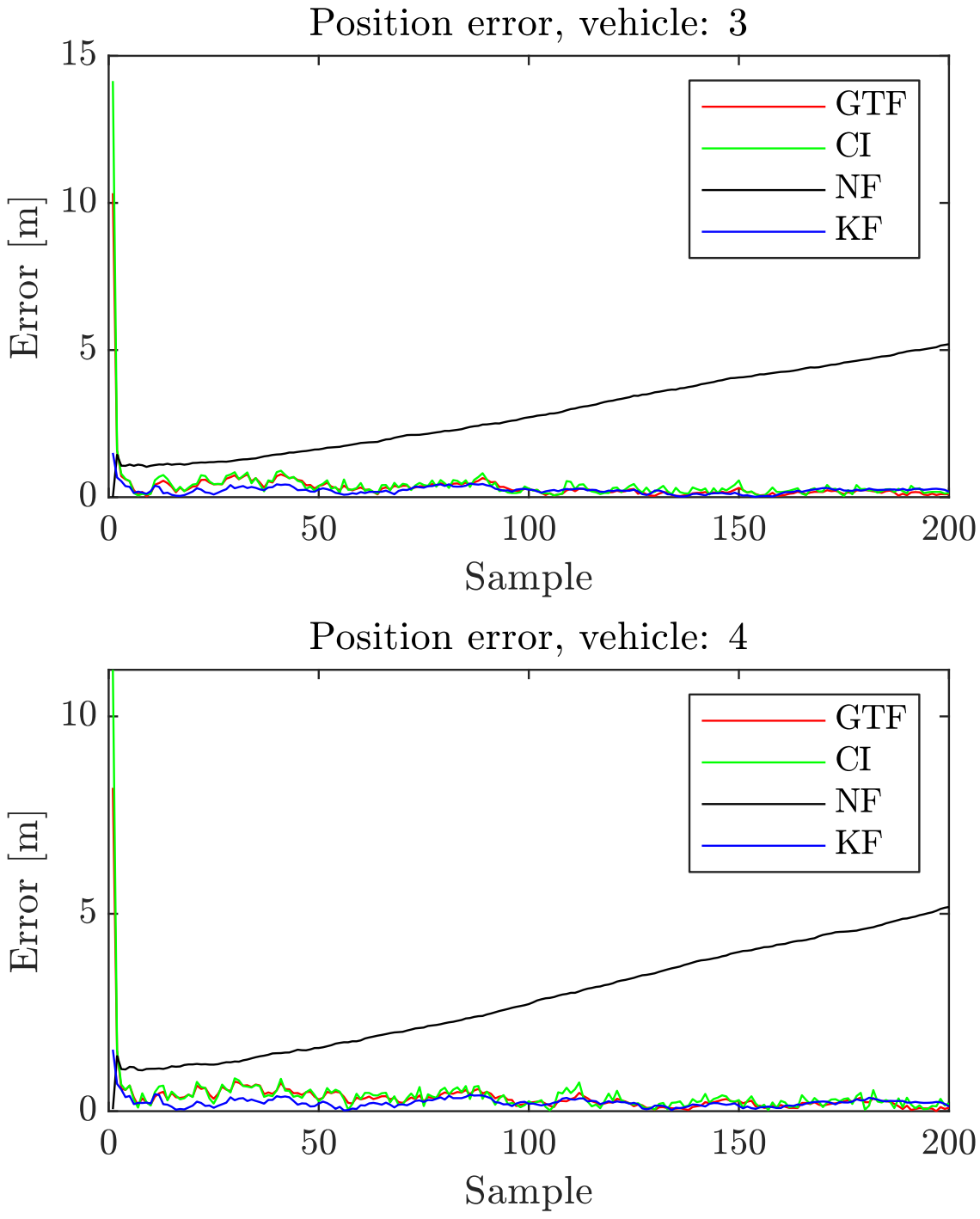

Figure 2.9: Position Error for Collaborative Vehicles 3, 4

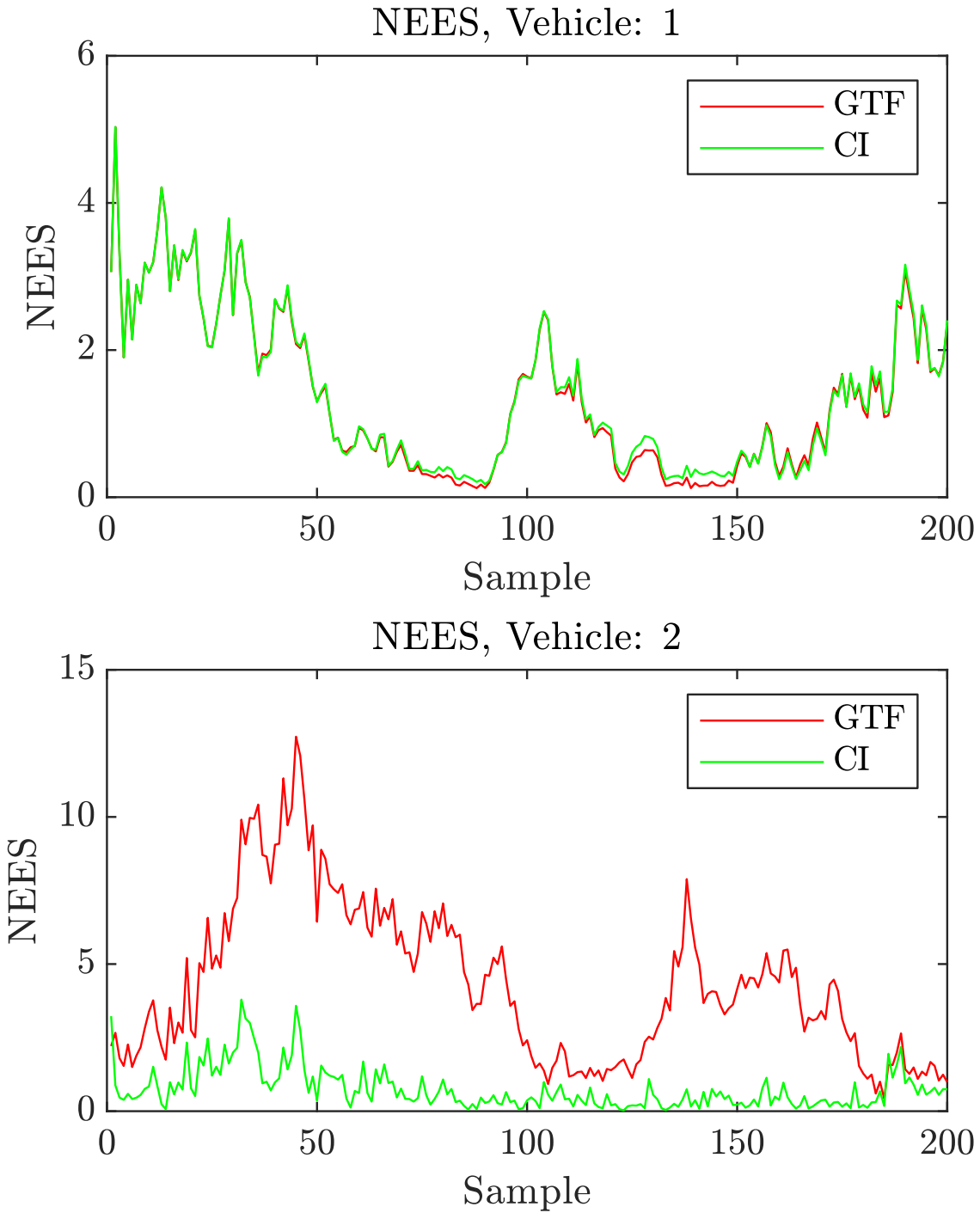

Figure 2.10: Normalized Estimation Error Squared for Collaborative Vehicles 1, 2

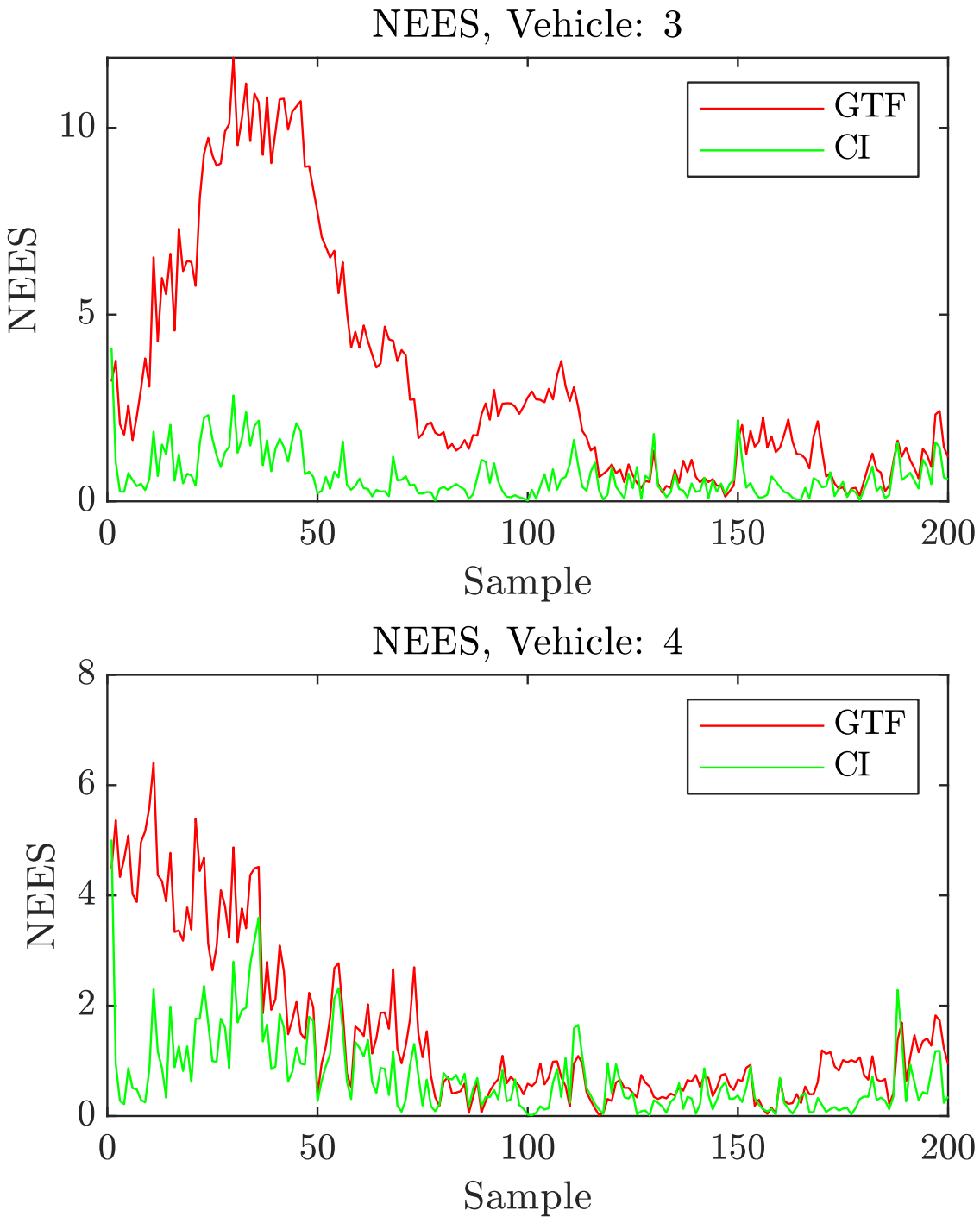

Figure 2.11: Normalized Estimation Error Squared for Collaborative Vehicles 3, 4

contributors. In this case, ARMAS should maintain at least $\binom{I}{C} = \binom{7}{3} = 35$ subfilters to ensure a proprioceptive filter is maintained. To mitigate the risk of losing overlapping state observability and double-counting, ARMAS should maintain at least F subfilters defined by (2.57).

$$F_{min} = \binom{I}{max[S, C]} \tag{2.57}$$

where I is the total number of sensors (onboard + offboard), S is the number of subfilters required to maintain overlapping observability, and C is the number of offboard measurements. Note the factorial growth of the required subfilters.

2.5.9 Chapter Summary.

This chapter summarized relevant methods required to tackle the collaborative all-source navigation problem. The chapter began by establishing notation conventions and reference frames used throughout this dissertation. Next, summaries of recursive Bayesian estimation, residual monitoring, inertial navigation, and all-source navigation were provided. Next, the resilient all-source ARMAS navigation framework and SAARM algorithms were discussed. An overview of stochastic observability methods was discussed in the context of navigation state analysis. We analyzed several current cross-correlation management techniques which can be used to support fusion of off-board collaborative sensor information. Lastly, we described NEES and its application for assessment of estimator credibility.

III. Real-time Trajectory Optimization for Collaborative Self-Localization in Random Aircraft Formations

3.1 Introduction

Persistent self-localization requires a resilient navigation solution, so it is critically important that we understand how to take advantage of performance benefits associated with collaborative localization and navigation [78]. Self-localization is the ability to accurately determine one's position accurately which is dependent on minimizing absolute and relative positioning errors. Early localization trajectory optimization methods were limited to offline estimation for a known reference trajectory [15], [32]. In [32], Helferty employed minimization of the $tr(CRLB_{\hat{\theta}})$ for trajectory optimization for offboard localization of a single target. Several recent techniques have been presented to optimize self-localization with a static network topology [3, 5, 39, 51, 55]. The relevance of localization optimization is underscored by recent developments in practical methods for maximizing localization performance under network communication resource constraints [57]. This chapter contributes a real-time heuristic method of trajectory generation for a single aircraft (referred to as the optimized node) to minimize relative positioning error in a wireless sensor network composed of a simulated formation of maneuvering aircraft.

With the proliferation of wireless sensor networks, there is an opportunity to collaboratively augment other aircraft in a degraded navigation environment. An anchor node can be defined as any node whose absolute location can be estimated accurately (e.g. GPS, highly reliable alternative navigation, etc.). In this scenario, simulated aircraft (anchor nodes) flying in a random dynamic formation are able to share their true position with nearby nodes over a wireless network. A single aircraft (optimized node) obtains noisy range measurements to each formation aircraft and combines this information with aircraft positions reported over a distributed wireless sensor network. Real-time trajectory guidance

is generated for the optimized node based on real-time minimization of relative positioning error. Since relative position error is based on the geometry between the optimized node and nearby formation nodes, this chapter introduces Self-Aligning Swarm (SAS), a real-time trajectory generation algorithm for self-localization.

In the following section, we review relevant background material which examines trade-offs associated with local and global cost minimization. We discuss the importance of algorithm scalability and conservation of resources given diminishing returns associated with capturing a global minimum which ultimately drove SAS towards local minimization. Next, we provide a derivation of the 2-D Cramér-Rao Lower Bound for a known variance which clarifies the importance of diverse angular coverage for reduction in relative positioning error. An introduction to the pseudolite cofactor matrix and comparison of SAS cost functions is performed and an anti-collision parameter is introduced. Next, we introduce the kinematics model for the simulated formation nodes and walk through an example of trajectory generation for a SAS-optimized node. Once the details of the simulation are explained, we provide the numerical results for a 10K trial Monte Carlo simulation. The results show the benefits of SAS optimization over a baseline formation trajectory. The chapter concludes with our overall assessment of SAS, potential applications for the algorithm, and recommendations for future work.

3.2 Self-Aligning Swarm (SAS) Algorithm for Self-localization

3.2.1 Background.

SAS was originally conceived as a distributed heuristic global optimization algorithm for self-localization. The initial goal was to optimize the real-time trajectory for single node self-localization across a complex, dynamic cost surface. At each time step, a CPU-intensive global search of the cost surface was performed. The complexity of this computation has motivated previous approximation methods such as the alpha shape for efficient estimation of localization performance [55]. A vector from the optimized

node's current position to the particle exhibiting the lowest cost was used to generate a guidance vector. This method required knowledge of the entire cost surface (which is likely unavailable) to guarantee a global minimum. The computation required to perform this search grows exponentially with search area. In a dynamically maneuvering formation of anchor nodes, the location of the global cost minimum can vary significantly, especially in the case of multiple competing minima. For example, there could be two or more minima fluctuating with approximately the same cost value in opposite directions from the optimized node. It would be counterproductive for a physical aircraft with momentum to expend resources to abruptly change direction to chase an ephemeral global minimum. There are diminishing returns associated with chasing a global minimum for self-localization.

For these reasons, a distributed heuristic local minimization algorithm was conceived with a small search area centered around the optimized node. To improve particle swarm efficiency, the size of the search area is no larger than the velocity and/or maneuvering capabilities of the optimized node. This means the search is constrained to positions the optimized node can actually achieve. This permits the designer to spawn a small, fixed number of particles for each heuristic search. By constraining the search area to the capabilities of the optimized platform, the local method does not guarantee a global minimum, nor does it guarantee the most direct trajectory to the minimum. However, the local method has a small, scalable computing burden and provides a stable, predictable trajectory to a dynamic self-localization optimum for reduction in relative position error and does not waste resources attempting to locate an unachievable solution.

3.2.2 2-D CRLB for Range Measurements in AWGN .

A node of interest receives range measurements corrupted with zero-mean independent bivariate Additive White Gaussian Noise (AWGN) from other nearby nodes in 2-D space. In the example, nearby nodes have unique positions separated by true distance D_n and each

range observation $\mathbf{r}_n$ is corrupted by zero-mean AWGN with magnitude σ. Ranging error $\mathbf{v}_n$ is known and clock synchronization is assumed.

$$\boldsymbol{r} = \boldsymbol{d} + \boldsymbol{v}, \text{ where } \boldsymbol{v} \hookrightarrow \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{S}) \tag{3.1}$$

where $\boldsymbol{r}, \boldsymbol{d}$, and $\boldsymbol{v}$ are vectors of dimension N × 1 with $n = \{1, 2, ..., N\}$ nearby nodes. $\boldsymbol{S} = \sigma^2 \boldsymbol{I_{2\times 2}}$ with σ = 5m with $\boldsymbol{\mu} = \mathbf{0}$. At each time step, i, the optimized node receives a range measurement R_n from each nearby node

$$\text{with nodes positioned at } \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}, \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}, ... \begin{bmatrix} x_n \\ y_n \end{bmatrix}.$$

Next we derive the Cramer-Rao Lower Bound (CRLB) for the self-localization position estimate $\hat{\Theta}$, where $\hat{\Theta}_i = \hat{x}_0$ and $\hat{\Theta}_j = \hat{y}_0$ using range measurements corrupted by zero-mean independent AWGN. The Fisher Information Matrix (FIM), $\mathbf{I}(\Theta)$, describes the observable information from the range measurements. The inverse of the FIM is the CRLB which provides a best-case estimate of the lower bound of the variance of the observable information (3.2).

$$\mathbf{I}(\hat{\Theta})^{-1} = CRLB_{\hat{\Theta}}. \tag{3.2}$$

To obtain the FIM, we examine the expected value of the derivative of the Probability Density Function (PDF) for each range measurement $\boldsymbol{r}_n$ with respect to our self-localization position estimate $\hat{\Theta}$ to understand how much information is available (3.3).

$$\mathbf{I}(\hat{\Theta}) = E\{[\frac{d}{d\hat{\Theta}} p(\mathbf{r}|\hat{\Theta})]^2\} = \int_{-\infty}^{\infty} \left(\frac{\frac{d}{d\hat{\Theta}}[p(\mathbf{r}|\hat{\Theta})]}{p(R|\hat{\Theta})} \right)^2 p(\mathbf{r}|\hat{\Theta}) d\mathbf{r}. \tag{3.3}$$

The pdf is

$$p(\mathbf{r}|\hat{\Theta}) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(\mathbf{r}-\hat{\Theta})^2}{2\sigma^2}}. \tag{3.4}$$

The partial derivative of (3.4) with respect to $\hat{\Theta}$ divided by the pdf from (3.4) is

$$\frac{\frac{d}{d\hat{\Theta}} p(\mathbf{r}|\hat{\Theta})}{p(\mathbf{r}|\hat{\Theta})} = \frac{\frac{d}{d\hat{\Theta}} \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(\mathbf{r}_n - \hat{\Theta})^2}{2\sigma^2}}}{\frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(\mathbf{r} - \hat{\Theta})^2}{2\sigma^2}}} = \frac{\mathbf{r} - \hat{\Theta}}{\sigma^2}.$$

When combined with (3.3) results in Fisher information shown in (3.5).

$$\begin{aligned} I(\hat{\Theta}) &= -\int_{-\infty}^{\infty} \left(\frac{\mathbf{r} - \hat{\Theta}}{\sigma^2}\right)^2 \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(\mathbf{r} - \hat{\Theta})^2}{2\sigma^2}} d\mathbf{r} \\ &= -\frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{\infty} \mathbf{r}^2 e^{\frac{-\mathbf{r}^2}{2}} d\mathbf{r} = \frac{1}{\sigma^2}. \end{aligned} \tag{3.5}$$

To obtain the Fisher Information matrix, we begin by taking the partial derivatives of each distance, D_n with respect to the node's location, x_0 and y_0.

$$I_{ij} = \frac{1}{\sigma^2} \sum_{n=0}^{N} \frac{d\mathrm{D}_n}{d\hat{\Theta}_i} \frac{d\mathrm{D}_n}{d\hat{\Theta}_j} \text{ where } \hat{\Theta}_i = x_0, \hat{\Theta}_j = y_0, \tag{3.6}$$

$$\text{and where } D_n = \sqrt{(x_0 - x_n)^2 + (y_0 - y_n)^2}).$$

$$\begin{aligned} \frac{d\mathrm{D}_n}{dx_0} &= \frac{d}{d\hat{\Theta}_i} \sqrt{(x_0 - x_n)^2 + (y_0 - y_n^2)} \\ &= \frac{1}{2}[(x_0 - x_n)^2 + (y_0 - y_n^2)]^{(-\frac{1}{2})}[2(x_0 - x_n)] \\ &= \frac{x_0 - x_n}{D_n} \end{aligned} \tag{3.7}$$

$$\begin{aligned} \frac{d\mathrm{D}_n}{dy_0} &= \frac{d}{d\hat{\Theta}_j} \sqrt{(x_0 - x_n)^2 + (y_0 - y_n^2)} \\ &= \frac{1}{2}[(x_0 - x_n)^2 + (y_0 - y_n^2)]^{(-\frac{1}{2})}[2(y_0 - y_n)] \\ &= \frac{y_0 - y_n}{D_n} \end{aligned} \tag{3.8}$$

We can avoid taking the log-likelihood by making use of basic trigonometry. Figure 3.1 shows that the partial derivatives of the distance between nodes are equivalent to the following trigonometric identities:

$$\cos(\phi_n) = \frac{x_0 - x_n}{D_n}$$

$$\sin(\phi_n) = \frac{y_0 - y_n}{D_n}$$

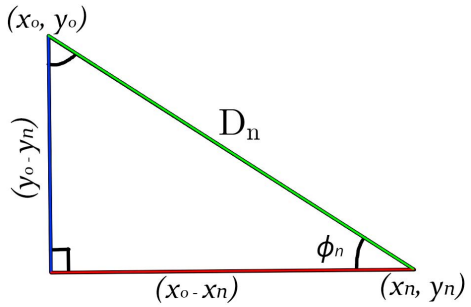

Figure 3.1: Visualization of Partial Derivatives of D_n

If we manipulate (3.6), we can show it is proportional to the expected value of the partial derivatives of the distance measurements. The elements of the Fisher information matrix (row i, column j) are

$$I_{ij} = \frac{N}{\sigma^2}\frac{1}{N}\sum_{n=0}^{N}\frac{d\mathrm{D}_n}{d\hat{\Theta}_i}\frac{d\mathrm{D}_n}{d\hat{\Theta}_j} \tag{3.9}$$

$$= \frac{N}{\sigma^2}E\left[\frac{d\mathrm{D}_n}{d\hat{\Theta}_i}\frac{d\mathrm{D}_n}{d\hat{\Theta}_j}\right]. \tag{3.10}$$

The resulting Fisher Information matrix is

$$\mathbf{I} = \frac{N}{\sigma^2}\begin{bmatrix} E[cos^2(\phi_n)] & E[cos(\phi_n)sin(\phi_n)] \\ E[sin(\phi_n)cos(\phi_n)] & E[sin^2(\phi_n)] \end{bmatrix} \tag{3.11}$$

accordingly, the Cramér-Rao Lower Bound is

$$Cov(\hat{\theta}) \geq \frac{\sigma^2}{N} \begin{bmatrix} E[cos^2(\phi_n)] & E[cos(\phi_n)sin(\phi_n)] \\ E[sin(\phi_n)cos(\phi_n)] & E[sin^2(\phi_n)] \end{bmatrix}^{-1} \tag{3.12}$$

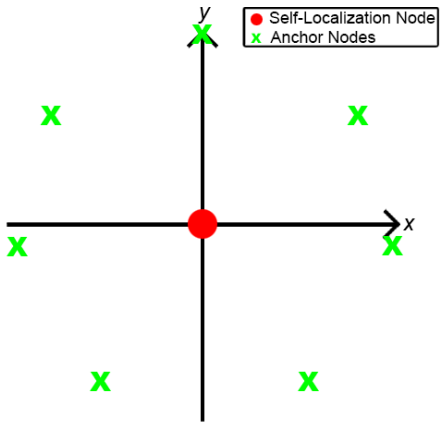

Figure 3.2: Optimal 2-D Self-Localization Geometry for $N = 7$ Anchor Nodes

The diagonal terms of the $CRLB_{\hat{\theta}}$ matrix define the dimensions of a 2-D covariance ellipse for relative positioning error. For example, minimum relative self-localization error for $N = 7$ anchor nodes is achieved when anchor nodes are evenly spaced and centered on the the self-localization node (See Figure 3.2). In this example, the diagonal terms of the FIM are

$$E[cos^2(\phi_n)] = E[sin^2(\phi_n)] \approx 0.5$$

and the off-diagonal terms are

$$E[cos(\phi_n)sin(\phi_n)] = E[sin(\phi_n)cos(\phi_n)] \approx 0.$$

The resulting CRLB is

$$COV(\hat{\theta}) \geq \frac{\sigma^2}{7}\begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}^{-1} = \frac{2\sigma^2}{7}\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}.$$

Assuming the anchor nodes remain static, the diagonal terms of the FIM will decrease if the Self-Localization node moves from its current position. This results in a larger value for the $tr(CRLB_{\hat{\theta}})$. This forms the motivation for minimization of the $tr(CRLB_{\hat{\theta}})$ for generating an optimized self-localization trajectory.

3.2.3 Pseudolite Cofactor Matrix and SAS Cost Function.

The definition of $CRLB_{\hat{\theta}}$ in Eq. 3.12 elucidates the importance of diverse angular coverage for reduction in relative positioning error variance. For a 2-D pseudolite-based self-localization problem, the design matrix, $\mathbf{P}$ for N nodes (Eq. 3.13), can be expressed in a similar manner as (3.7) and (3.8).

$$\boldsymbol{P} = \begin{bmatrix} \frac{x_1 - x_0}{r_1} & \frac{y_1 - y_0}{r_1} \\ \frac{x_2 - x_0}{r_2} & \frac{y_2 - y_0}{r_2} \\ \cdots & \cdots \\ \frac{x_N - x_0}{r_N} & \frac{y_N - y_0}{r_N} \end{bmatrix} \text{ where } r \text{ is defined by Eq. 3.1.} \tag{3.13}$$

The pseudolite cofactor matrix $\mathbf{\Sigma_c}$ for this example can be computed using a least squares method shown in (3.14)

$$\mathbf{\Sigma_c} = (\mathbf{P}^T\mathbf{P})^{-1} = \frac{\sigma^2}{N}\begin{bmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{yx} & \sigma_y^2 \end{bmatrix}. \tag{3.14}$$

Minimization of the $tr(\mathbf{\Sigma_c})$ (Eq. 3.15) is equivalent to minimizing the $tr(CRLB_{\hat{\theta}})$.

$$tr(\mathbf{\Sigma_c}) = \frac{\sigma^2}{N}(\sigma_x^2 + \sigma_y^2). \tag{3.15}$$

It is assumed that the initial starting position $[x_0, y_0]^T$ of the optimized node is known and is used to initialize an iterative nonlinear self-localization position estimator which

relies on MATLAB's $fitlnm$ nonlinear model fit function to perform a self-localization estimate. For all $i > 1$, the t_{i-1} position estimate is used to initialize the self-localization estimator. The cost function employed in the SAS algorithm is composed of trajectory optimization with an anti-collision function (3.16).

$$SAS_{Cost} = tr(\mathbf{\Sigma_c}) + A^{\sum_1^N r_n^{-1}}. \tag{3.16}$$

where A is a desired anti-collision avoidance factor, N is the total number of nearby formation nodes, and $\boldsymbol{r}_n$ is a noisy range measurement from node n.

The avoidance term in the cost function is approximately 1 until the optimized node is in proximity of a formation node. A determines the gradient of this anti-collision function. In this simulation, A was chosen to mitigate incursions inside 100 meters of other nodes. SAS minimizes Eq. 3.16 at each time step to determine an optimal trajectory for the optimized node.

3.2.4 Simulation.

The general shape of a surface defined by the $tr(CRLB_{\hat{\theta}})$ for a formation of networked nodes is convex, with the global minimum typically existing somewhere near the center of the formation network. Since self-localization performance is based primarily on angular coverage from anchor nodes, there can be significant benefits to trajectory optimization with suboptimal formation topology.

In the scenario illustrated in Figure 3.3, there is a single optimized node and $N = 7$ anchor nodes in a random, dynamic formation. Figure 3.3 shows the results of this initialization with a single optimized node and N formation nodes visible on a 2-D plane at $t_0 = 0.0$ sec. N nodes provide their absolute positions to a optimized node over a simulated digital wireless network. The single optimized node is able to collect noisy range measurements to each formation node with $\sigma_{range} = 5$ m (detailed explanation presented in Section 3.2.3). $10,000$ Monte Carlo trials were performed lasting a duration of $T_{trial} = 150$

sec each. At the beginning of each run, all of the nodes are initialized at random, normally-distributed 2-D locations on a square 400 km^2 grid.

To simulate a randomly maneuvering formation of fixed-wing aircraft travelling in the same direction, the formation nodes are initialized with a stable 2-D velocity V_0 = 100 m/s initially in the $+X$ direction with a Gaussian random angular perturbation defined by Eq. 3.17.

$$\Theta_i = 0.8\Theta_{i-1} + 0.2\phi. \tag{3.17}$$

where time step size t_i is 1 second, Θ is the current 2-D velocity angle and ϕ is a normally distributed random angle between [0, 2π] radians generated for each formation node every time step t_i. The effect of Eq. 3.17 is visible in Figure 3.4.

In the same manner as the formation nodes, the optimized node is initialized at a random 2-D location on the 400 km^2 grid. Noisy range measurements are performed between the optimized node and each formation node at each time step. MATLAB's 'fit nonlinear model' function $fitnlm$ is used to obtain an iterative least-squares position estimate for the optimized node's absolute 2-D position using the noise-corrupted range measurements and true absolute positions of the nearby nodes. Detailed explanations of the range measurements and absolute position estimate methods are presented in Section 3.2.3.

The optimized node is infinitely maneuverable with a maximum velocity of 250 m/s. The trajectory of the optimized node is based on a particle swarm search of size 10 particles inside a local circular area of radius 250 m centered on the location of the optimized node. The optimized node trajectory is established by following the local gradient of the cost function at each time step. Once a local minimum is captured inside the search area, the optimized node follows the surface gradient to the minimum and maneuvers as necessary

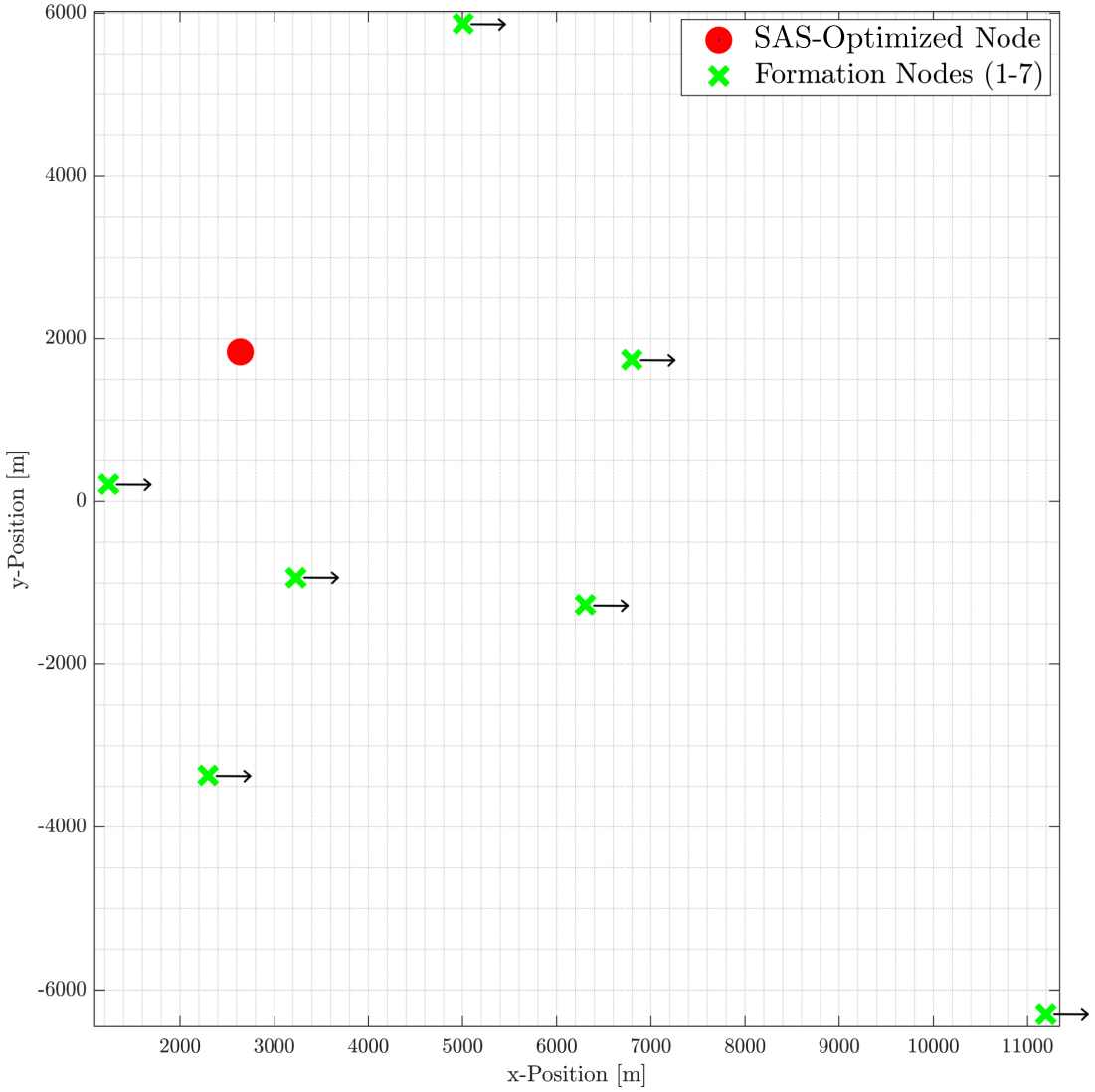

Figure 3.3: SAS Initialization with Random Node Locations and Initial Trajectory, $t_0 = 0.0$ sec

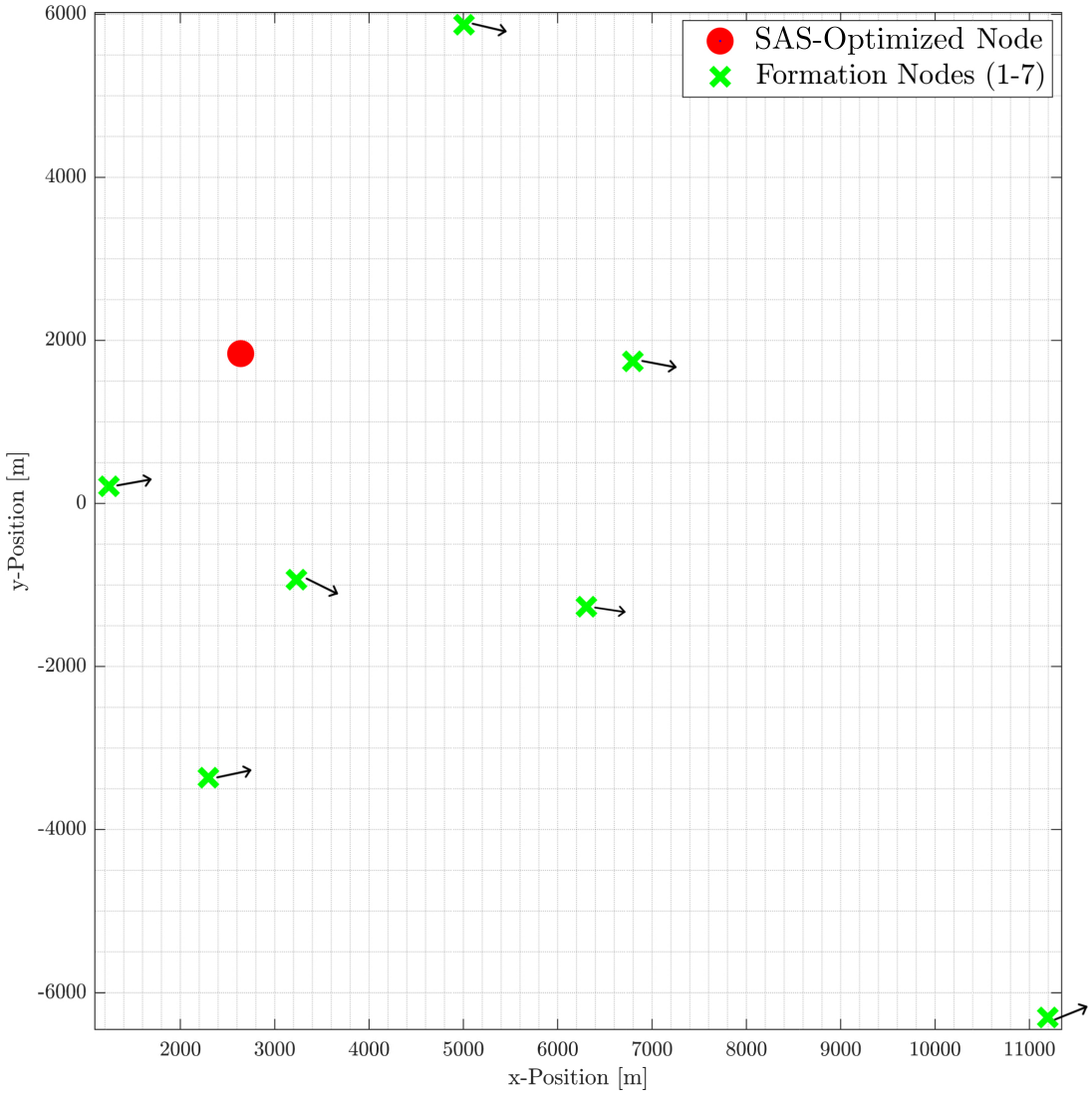

Figure 3.4: SAS Formation Node Trajectories with Angular Perturbations, t = 1.0 sec

to maintain the local minimum. The optimized node will continue to follow the surface gradient towards a local minimum, resulting in a real-time optimized self-localization trajectory.

Figure 3.5 shows the contour of the particle swarm cost function given by Eq. 3.16 for the configuration shown in Figure 3.3 from the perspective of the optimized node. The peaks resulting from the anti-collision term visible in Figure 3.5 indicate where the formation nodes are located. These are the same nodes visible in Figure 3.3. Figure 3.6 shows the contour surface of $tr(CRLB_{\hat{\theta}})$ without the anti-collision term. It is apparent that the locations of the minima of the $tr(CRLB_{\hat{\theta}})$ are unaffected by the anti-collision term which ensures that minimization of the cost function is equivalent to minimization of the $tr(\Sigma_c)$ and $tr(CRLB_{\hat{\theta}})$, from Equations 3.12 and 3.14.

The optimized node's current trajectory follows the gradient of the cost surface in Figure 3.5 towards the minimum value of the surface and is identified by a vector arrow. Section 3.2.2 presents a derivation for the lower bound of the 2-D error variance. A proof for the convergence of the bounds of total relative localization error to the minimization of the $tr(CRLB)$ is presented in [6].

As the simulation progresses, the heuristic search is repeated at each time step to establish a new local minimum for self-localization trajectory optimization. Once the optimized node arrives at the local minimum, its velocity will stabilize with the surrounding formation and the optimized node will maneuver to maintain optimal self-localization geometry within the local search area. At each time step, position residuals are recorded for the optimized node and position MSE is recorded for the duration of the run. The results for a 10K-trial Monte Carlo simulation are discussed in Section 3.3.

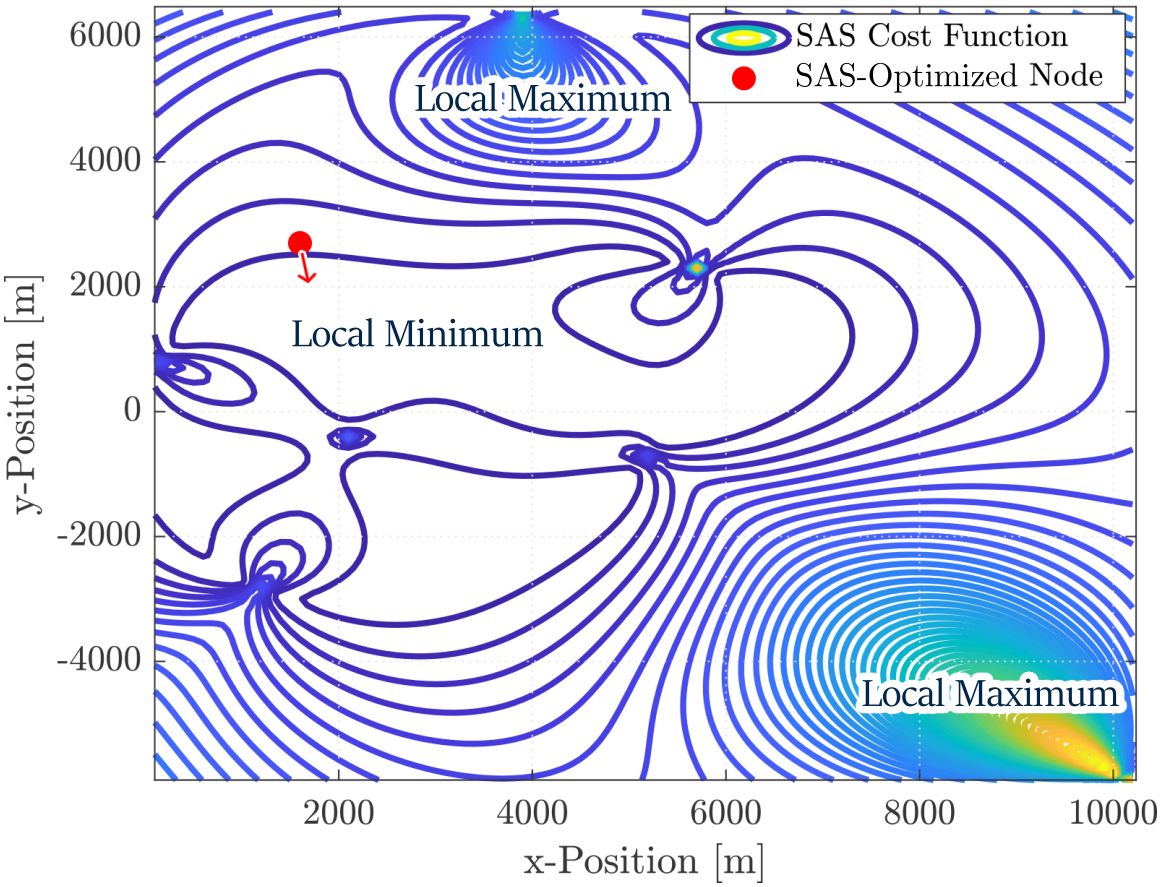

Figure 3.5: Initial Node Trajectory Towards Local Minimum of Cost Function, $t_0 = 0.0$ sec

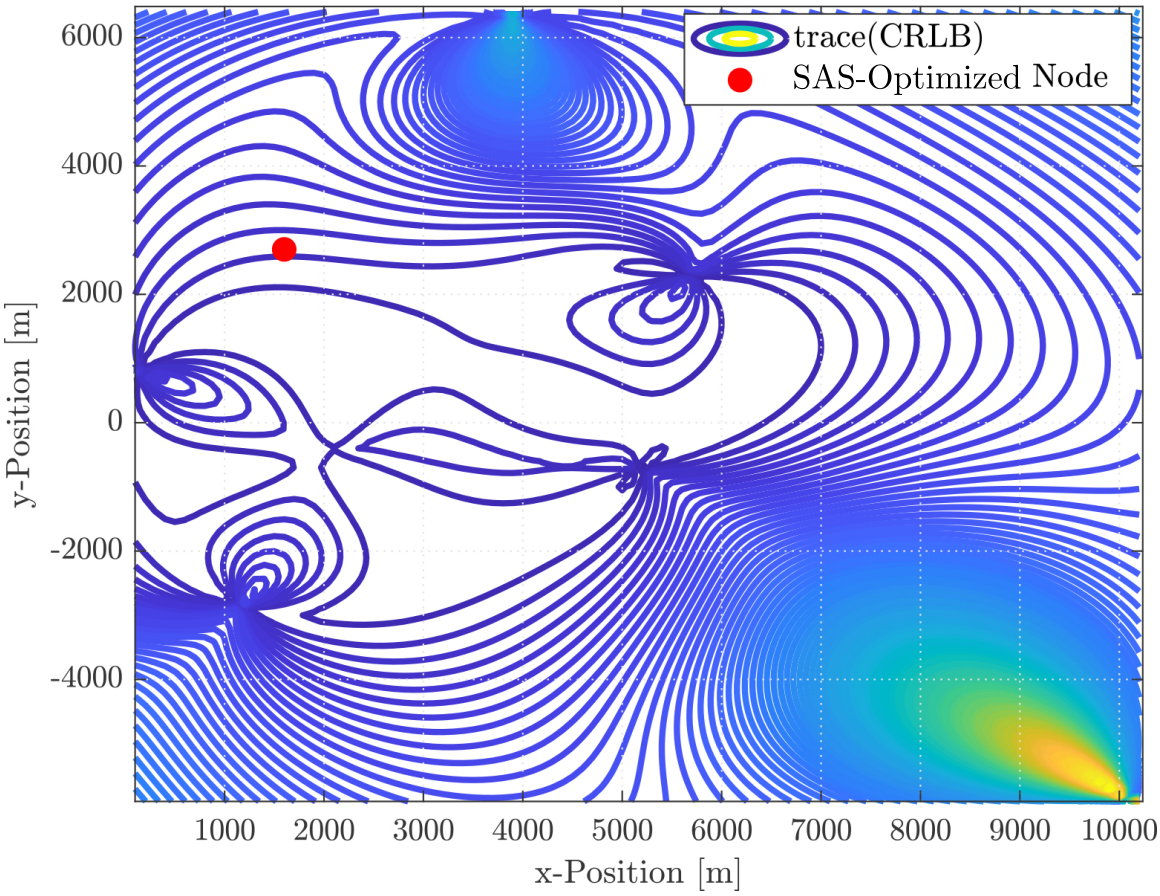

Figure 3.6: Contour of $tr(CRLB_{\hat{\theta}})$ for Formation Node Network, t_0 = 0.0 sec

3.3 Numerical Results

The estimated mean position MSE for the SAS-optimized node and each of the 7 non-optimized formation nodes are shown for 10K trials in Table 3.1. To develop a baseline for comparison, localization estimates for the formation nodes were performed using the same noisy ranging technique described in Section 3.2.3. The formation nodes perform a 'default' behavior by generally maintaining the same relative formation position for the duration of the simulation. This comparison quantifies the difference between mean position MSE observed from maintaining formation and from following the SAS-optimized trajectory. Figure 3.8 shows the mean position MSE distributions for each of the 7 non-SAS formation nodes have nearly the same distribution for 10,000 trials. A t-test for significant differences in mean variances indicates the variance of position MSE for the SAS-optimized node is smaller than that of formation nodes with a p-value < 0.0001. The reduction in variance is shown by a violin-plot comparison of the position MSE distributions for 10K Monte Carlo trials shown in Figure 3.7.

Table 3.1: Mean Position MSE for 10K Trials, T_{trial} = 150 sec, σ_{range} = 5 m

Type/Behavior	$\hat{\mu}_{MSE}$	$\hat{\sigma}_{MSE}$
Formation Nodes	**4.06 m**	**0.981 m**
SAS-Optimized Node	**3.41 m**	**0.184 m**

The overall reduction in mean position MSE and variance can be explained with an example. If a node is randomly spawned at an unfavorable formation position, it is expected to have a large CRLB (and position MSE) compared to a node spawned in a more favorable position within the formation. In both situations, a SAS-optimized node will follow a trajectory to minimize the $tr(CRLB_{\hat{\theta}})$ which results in lower position MSE. In contrast, a

formation node simply flies along with the formation with kinematics governed by Eq. 3.17 and does not optimize its position MSE. This difference in behavior results in the reduced mean and variance for the SAS-optimized mean position MSE distribution.

A comparison of Normal-Gaussian distribution models fit to the SAS-optimized and Formation Node 1 distributions is visible in Figure 3.9. The mean position MSE for the SAS-optimized node approaches a Normal-Gaussian distribution which is a property not shared by the formation trajectory nodes. Normally-distributed position error is desirable for subsequent use with Bayesian estimators which often make this simplifying assumption regardless of the actual error distribution.

3.4 Chapter Summary and Future Work

This chapter presents a novel real-time heuristic method of trajectory optimization for a single SAS-optimized aircraft to minimize self-localization error in proximity to a formation of randomly maneuvering aircraft. SAS is a compact, scalable heuristic method to optimize collaborative self-localization performance for a single aircraft in a random formation. SAS is shown to be a useful method for dynamic position error minimization. A 10K-trial Monte Carlo simulation demonstrated SAS-optimized flight path trajectory exhibits reduced mean position MSE and reduced variance for position MSE. The simulation also shows the distribution of position MSE for the SAS-optimized aircraft approaches normality whereas any other random formation aircraft exhibited a heavily-skewed error distribution. A potential practical application for SAS is implementation on a potentially GPS-denied aircraft whose goal is to minimize Inertial Navigation System drift in a collaborative network composed of all-source contributors. Recommended future work includes investigation of multiple cooperative anchor nodes for optimized localization performance of off-board aircraft on a mission-dictated trajectory in a dynamic environment.

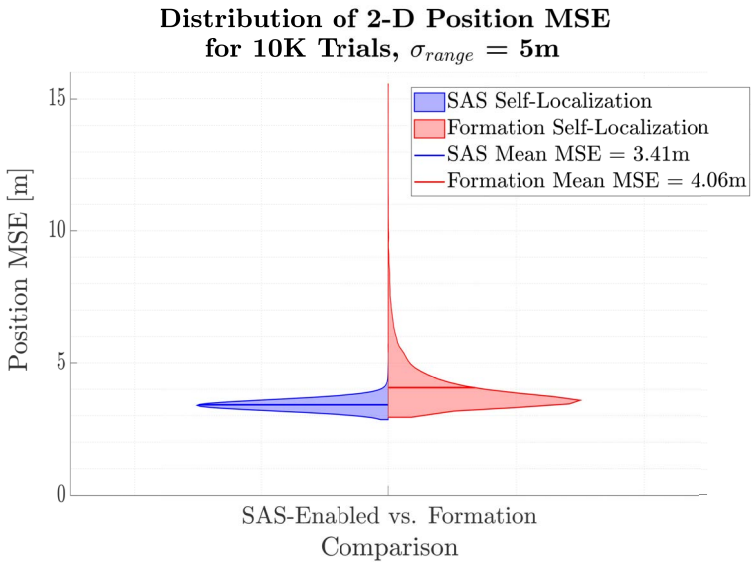

Figure 3.7: Distribution of Mean Position MSE for 10K Trials, T_{trial} = 150 sec, σ_{range} = 5 m

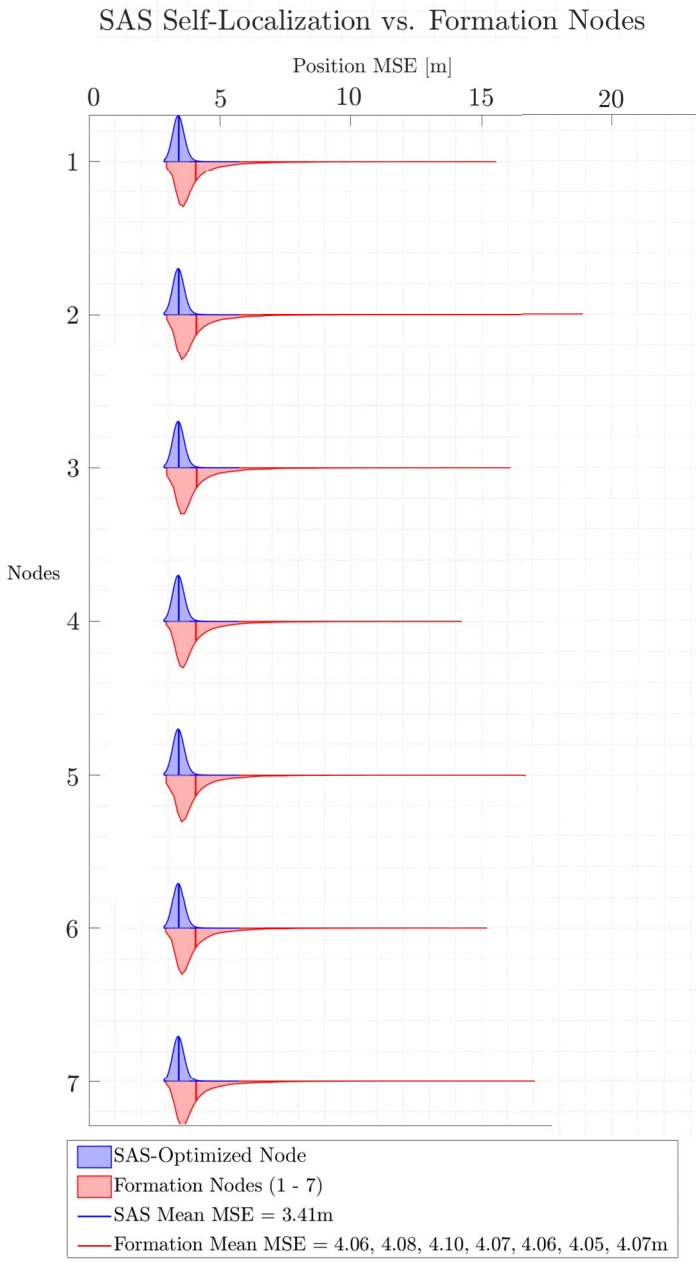

Figure 3.8: Comparison of Mean Position MSE Distributions for 10K Trials, T_{trial} = 150 sec, σ_{range} = 5 m

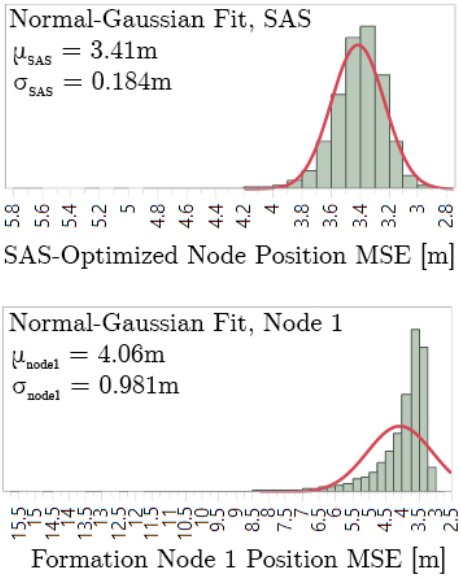

Figure 3.9: Comparison of Normal-Gaussian Models Fit to SAS-Optimized and Formation Node 1 Mean Position MSE Distributions for 10K trials, T_{trial} = 150 sec, σ_{range} = 5 m

IV. Swarm Control for Autonomous Navigation Support

4.1 Introduction

With modern computing resources, concurrent complex calculations can be performed in real time and can be harnessed to optimize vehicle behavior for distributed tasks such as swarm control. However, distributed algorithms must still employ a scalable topology to be practically simulated and deployed. Scalability is particularly pertinent for large swarms. This chapter focuses on Swarm Control for Autonomous Navigation Support (SCANS) operating in a distributed simulation environment [27]. The application for SCANS is real-time swarm-based navigation support in Global Navigation Satellite System (GNSS) degraded and/or denied regions. In these areas, GNSS signals are either unavailable or untrustworthy as a result of signal interference due to downlink jamming, spoofing, or environmental features. Vehicles not equipped with all-source navigation sensors (e.g. magnetic, visual, etc.) are subject to inertial integration drift in these types of environments. Navigation support vehicles can be used to provide persistent autonomous multilateration for nearby vehicles tasked with missions (Fig. 4.1). Given sensor payload, fuel and cost constraints associated with small Unmanned Aerial Systems (sUAS), it is desirable to federate costly all-source sensor payloads from mission payloads, especially if the mission vehicles must be attritable. For example, an optically pumped magnetometer costs approximately $20k [14]. The mission vehicles benefit from persistent surveillance network multilateration support.

This chapter examines Swarm Control for Autonomous Navigation Support (SCANS)– a distributed, importance-weighted swarm control scheme for autonomous navigation support. This chapter builds on previous work [26] and [27] by utilizing the SAS simulation environment to provide a noise-corrupted network with realistic ranging and positioning for navigation support vehicles. The primary goal of this chapter is to demonstrate the

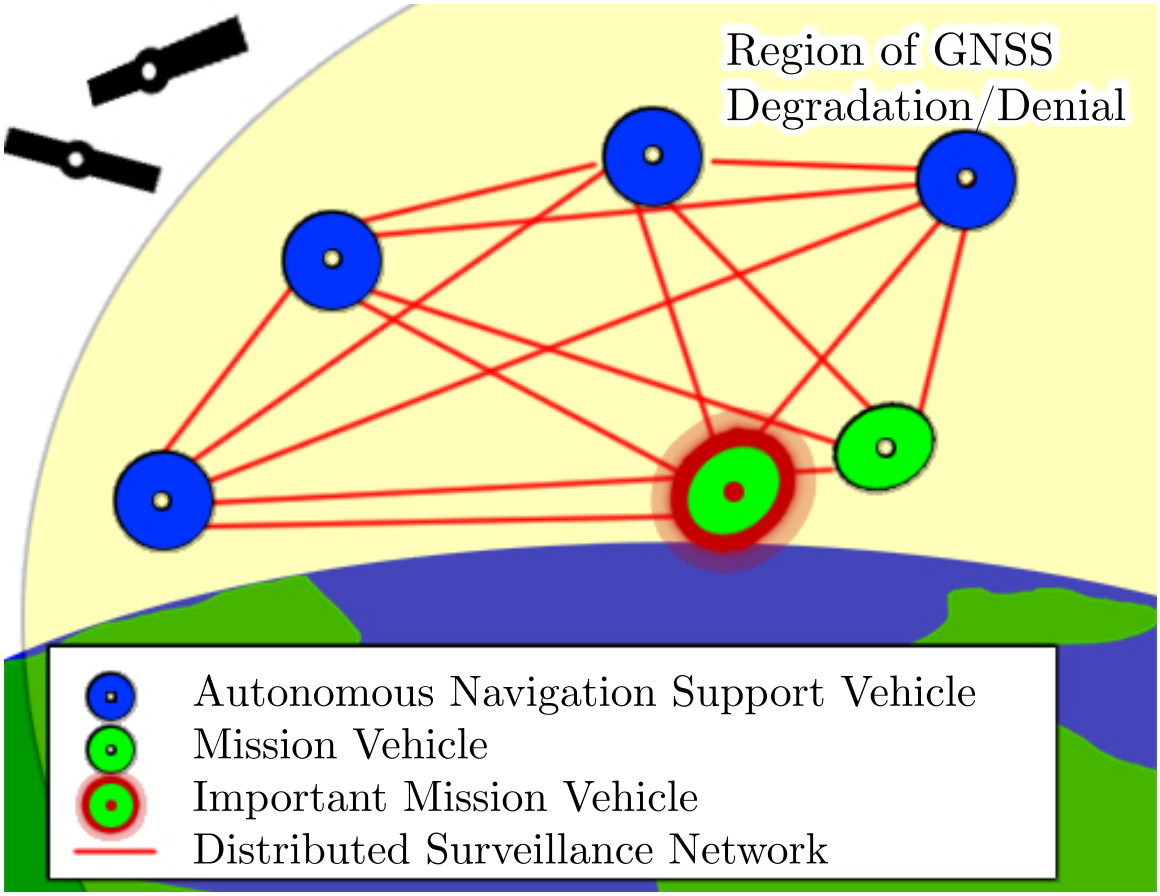

Figure 4.1: Importance-Weighted Navigation Support using Surveillance Network

full selective, importance-weighted navigation support capabilities of SCANS within the context of the SAS simulation environment. Four separate 10K Monte Carlo trials, encompassing over 1600 hours of simulated flight time, are performed for a 10-vehicle network with mission vehicle weights varying from 0 to 100. We present results in terms of the 2-D position RMSE for each mission vehicle. Monte Carlo trial results show optimized RMSE for simultaneous multi-vehicle support. Mission vehicles with prominent importance (relative to the size of the swarm) achieve RMSE distributions that approach normality. This chapter demonstrates results of large swarms (> 50 vehicles) with an overview of various stable convergent SCANS vehicle formations.

In the following section, we examine previous work which resulted in the development of SCANS. We examine the development of the SCANS algorithm based on distributed greedy minimization and highlight key decisions made to enable scalable consensus-like behavior. For context, we summarize the concept of magnetic anomaly field navigation and recent real-world test results. During our analysis, we introduce the importance weighting matrix which allows a distributed cooperative greedy algorithm to act in a non-greedy manner and optimize offboard vehicle navigation. Next, we introduce the mission aircraft kinematics models and explain the quasi-cooperative control scheme for navigation support aircraft. Once the details of the simulation are explained, we provide position RMSE for supported mission vehicles in four separate 10K trial Monte Carlo simulations. The weights assigned to each mission vehicle vary from 0 to 100. Our results show that distributed greedy optimization results in weighted consensus-like optimization for various subsets of mission vehicles. The chapter concludes with our overall assessment of SCANS, potential applications for the algorithm in the realm of collaborative navigation, and recommendations for future work.

4.2 Related Work

4.2.1 Decentralized-Local vs. Centralized-Global Optimization.

Dynamic global optimization for a networked formation of anchor vehicles is quite complex and has motivated simplifying approximation techniques like the alpha-shape [55]. In the 1990s, any approach to trajectory optimization for relative localization was limited to post-processed global algorithms [15, 32]. In [32], the author employed minimization of the trace of the Cramér-Rao Lower Bound (CRLB), for iterative offline trajectory optimization to localize a single moving target with a known trajectory. As long as anchor locations can be estimated accurately, the minimization of the trace of the CRLB is an effective technique [32] and forms the core optimization strategy of SCANS. More recently, there have been several successful strategies presented to optimize self-localization with static anchor locations [3, 5, 39]. All of these approaches relied on a global optimization technique for localization with known static anchor locations.

Globally optimized trajectory control for single vehicle self-localization requires a centralized architecture which is accompanied by increased communications overhead and limited scalability [27, 71]. Coordinated dynamic multi-agent navigation support for non-cooperative mission vehicles is even less tractable for global optimization. In a dynamically maneuvering formation of anchor nodes, the location of the global cost minimum can vary significantly, especially in the case of multiple competing minima. For example, there could be two or more minima fluctuating with approximately the same cost value in opposite directions from the optimized node. It would be counterproductive for a physical aircraft with momentum to expend resources to abruptly change direction to chase an ephemeral global minimum. There are diminishing returns associated with chasing a global minimum for self-localization.

To guarantee a global minimum cost, the positions of all vehicles must be available by a central agent and used to form an optimal set of multilateration position solutions

for each mission vehicle in the surveillance network. Once the solutions are formed, one could solve for a set of navigation support vehicle locations which would guarantee a global optimum for multilateration accuracy at that instant. Once solved, control vectors could be formed from each navigation support vehicle's current location to the location associated with a global minimum. The navigation support vehicles would each move in the direction of this guidance solution. Since the mission aircraft trajectories are not known *a priori*, this optimization is only valid for the current instant and would need to be performed recursively. Centralized processing of all vehicle trajectories would be required for deconfliction and is expected to require significant communications overhead.

For the aforementioned reasons, a distributed heuristic local minimization approach was conceived with a small search area centered around the navigation support vehicle [26]. A particle swarm was selected to sample a locally-constructed cost surface. The heuristic computing burden is established by the size of the search area, which is no larger than the velocity and/or maneuvering capabilities of the navigation support vehicle. This means the search is constrained to positions the vehicle can actually reach in the next time step [27]. This permits the particle swarm architect to generate a fixed number of particles for each heuristic search which establishes a theoretical ceiling for the computing burden. Most importantly, this computing burden scales linearly with the number of agents. By constraining the search area to the vehicle's capabilities, this local method does not guarantee a global minimum, nor does it guarantee the most direct trajectory to the minimum [27]. However, SAS has a small, scalable computing burden, provides a stable, predictable trajectory to a dynamic optimum and does not waste resources attempting to locate an unreachable trajectory.

4.2.2 Magnetic Navigation.

Magnetic navigation has been demonstrated as a persistent and accurate absolute positioning source [12]. Unlike other feature-based techniques like vision-based

navigation, variations in the Earth's magnetic field are ubiquitous and can be continuously sampled over the entire surface of the planet. For example, salient visual features are probably sparse over large bodies of water or heavily forested regions. Star-trackers are limited by background noise which is highly dependent on the diurnal cycle [53]. Since magnetic measurements are based on the summation of all magnetic field components at any given point, magnetic reference map detail is greatest near the Earth's surface. Aircraft are uniquely suited for magnetic anomaly navigation because they are able to sample large, spatially diverse areas while maintaining relative proximity to the Earth's surface [12]. Magnetic anomaly navigation can be performed passively and is resistant to interference [14]. Once a reference map is created based on a magnetic survey, a magnetic anomaly map can be constructed by a vehicle equipped with a sensitive magnetometer [34]. "A magnetic anomaly map is the difference between the scalar magnetic field magnitude at any point in space and the predicted magnetic field from a reference model" [14]. Typically, a particle filter or similar non-Gaussian estimator is used for trajectory correlation and subsequent localization. A flight test was performed in 2015 [14] over Louisa, VA using a single aircraft equipped with an optically pumped magnetometer which achieved a horizontal RMSE of 13.1 m under quiet atmospheric conditions. A horizontal RMSE of 15 m was predicted under non-stable (stormy) atmospheric conditions. These results were used to configure the absolute 2-D localization accuracy of the magnetic navigation support vehicles simulated in the next section.

Terrestrial magnetic field anomaly navigation continues to show promise as means of absolute positioning. For example, passively detected unique magnetic field anomalies associated with known geographic locations or geomagnetic "signboards" were recently studied as a means to validate GPS accuracy in the presence of suspected jamming or spoofing [37]. In 2019, unique magnetic anomaly data were gathered by a vehicle equipped with a magnetometer driving on public roads in a developed area near Eglin AFB, FL. The

magnetometer was used to sample each of the four geographic magnetic field anomaly locations and store the geomagnetic "signboard" signature data. Three days later, the same configuration was used to measure magnetic field anomalies along a continuous driving route and was able to positively correlate measurements with each correct landmark. The authors reported strong Pearson's correlation coefficients based on 95% confidence intervals for correct localization for each signboard.

4.3 Swarm Control for Autonomous Navigation Support (SCANS) Algorithm for Swarm Control

4.3.1 Introduction.

In a GNSS-denied environment, an autonomous swarm of magnetic navigation support vehicles and a separate set of mission vehicles broadcast their positions over a 10-vehicle surveillance network. Autonomous trajectory control is performed for a set of navigation support vehicles using the distributed SCANS algorithm to provide navigation support of a set of mission vehicle(s). Operating on each navigation support vehicle, SCANS uses a weighting matrix to proportionally distort the cost surface in favor of the mission vehicle(s)' localization by incentivizing navigation support vehicles to sacrifice their own accuracy in favor of important offboard mission vehicle(s). This section describes the position-based localization algorithm, the development of the cost function, and the implementation of the SCANS simulation.

4.3.2 Localization Algorithm.

To localize, a mission vehicle performs 2-D range measurements corrupted with zero-mean independent bivariate Additive White Gaussian Noise (AWGN) to navigation support vehicles. Sensor positions are transformed into an East-North coordinate system with a predefined origin. Nearby vehicles have unique positions separated by true distance D_n and each range observation $\underline{r}_n$ is corrupted by zero-mean AWGN with magnitude σ (4.1). Ranging error $\underline{v}$ is known and clock synchronization is assumed.

$$\underline{r} = \underline{d} + \underline{v}, \text{ where } \underline{v} \hookrightarrow \mathcal{N}(\underline{\mu}, \underline{\underline{S}}). \tag{4.1}$$

where $\underline{r}$, $\underline{D}$, and $\underline{v}$ are vectors of dimension N $\times$ 1 with $n = \{1, 2, ..., N\}$ nearby vehicles. $\underline{\underline{S}} = \underline{\sigma}^2 \underline{\underline{I}}$ with σ = 5m and $\mu = \underline{0}$. At each time step, i, a mission vehicle receives a range measurement r_n from each nearby navigation support vehicle with positions reported as

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} + \underline{w}_1, \begin{bmatrix} x_2 \\ y_2 \end{bmatrix} + \underline{w}_2, ... \begin{bmatrix} x_N \\ y_N \end{bmatrix} + \underline{w}_N,$$

$$\text{where } \underline{w}_n \hookrightarrow \mathcal{N}(\underline{\mu}_{mag}, \underline{\underline{Q}}_n) \tag{4.2}$$

for navigation support vehicles and

$$\text{and } \underline{w}_n \hookrightarrow \mathcal{N}(\underline{\mu}_{relnav}, \underline{\underline{\Sigma}}_c) \tag{4.3}$$

for mission vehicles where each $\underline{w}_n$ is a vector of dimension 2 $\times$ 1 for $n = \{1, 2, ..., N\}$. $\underline{\underline{Q}}_n = \sigma^2_{mag} \underline{\underline{I}}$ with σ_{mag} = 14.5 meters and $\underline{\mu}_{mag} = \underline{0}$. $\underline{\underline{\Sigma}}_c$ is the cofactor matrix defined by (4.10) and $\underline{\mu}_{relnav} = \underline{0}$. At each time step, i, a noisy range measurement r_n is performed between a mission vehicle and each nearby navigation support vehicle.

It is assumed that the initial starting position $[x_0, y_0]^T$ for each vehicle is known and is used to initialize an iterative nonlinear self-localization position estimator which relies on MATLAB's $fitnlm$ nonlinear model fit function to perform an instantaneous "snapshot" self-localization estimate without Bayesian estimation. For all $i > 1$, the previous t_{i-1} mission vehicle position estimate is used to initialize the current time step t_i self-localization estimator.

At each time step t_i, 2-D position error is recorded for each mission vehicle in the network. We use mission vehicle root-mean-square (RMS) position error as the primary metric to assess algorithm performance. The results of four separate 10K Monte Carlo trials are presented in the results section to demonstrate the weighted SCANS algorithm.

4.3.3 Pluggable Cost Function.

The SCANS cost function is based on minimization of the trace of the Cramér-Rao Lower Bound, $tr(CRLB)$, for ranging with known noise in a pseudolite network. A more detailed derivation for single vehicle localization is provided by [27]. The Fisher Information Matrix (FIM) is defined as the inverse of the Cramér-Rao Lower bound (4.4) and indicates how much information the noisy range measurements contain for localization.

$$\underline{\underline{I}}(\hat{\Theta})^{-1} = CRLB_{\hat{\Theta}} \tag{4.4}$$

To derive the CRLB for a signal in AWGN, we examine the log-likelihood for each observation, r_n, with respect to the distance to each vehicle, n, $d_n(\underline{\Theta})$, with dependency on relative geometry to understand how much information is available. A detailed derivation using the log-likelihood function is presented in [26].

The resulting Fisher Information matrix is

$$\underline{\underline{I}} = \frac{N}{\sigma^2}\begin{bmatrix} E[cos^2(\phi_n)] & E[cos(\phi_n)sin(\phi_n)] \\ E[sin(\phi_n)cos(\phi_n)] & E[sin^2(\phi_n)] \end{bmatrix}. \tag{4.5}$$

Accordingly, the Cramér-Rao Lower Bound is

$$Cov(\hat{\theta}) \geq \frac{\sigma^2}{N}\begin{bmatrix} E[cos^2(\phi_n)] & E[cos(\phi_n)sin(\phi_n)] \\ E[sin(\phi_n)cos(\phi_n)] & E[sin^2(\phi_n)] \end{bmatrix}^{-1}. \tag{4.6}$$

For a 2-D pseudolite-based self-localization problem, the design matrix, $\underline{\underline{P}}$ for N vehicles can be expressed as (4.7)

$$\underline{\underline{P}} = \begin{bmatrix} \frac{x_1-x_0}{r_1} & \frac{y_1-y_0}{r_1} \\ \frac{x_2-x_0}{r_2} & \frac{y_2-y_0}{r_2} \\ \cdots & \cdots \\ \frac{x_N-x_0}{r_N} & \frac{y_N-y_0}{r_N} \end{bmatrix} \text{ where } \underline{r} \text{ is defined by (4.1).} \tag{4.7}$$

A diagonal weighting matrix enables the designer to set individual importance values for each vehicle in the swarm (4.8). The weighting matrix is designed to distort the cost surface in favor of a particular set of vehicles.

$$\underline{\underline{W}} = \begin{bmatrix} w_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & w_N \end{bmatrix} \text{ for 1 to } N \text{ vehicles} \tag{4.8}$$

For navigation support vehicles, a weight of 1 is assigned. Off-diagonal values are always set to zero. A user configurable weight of zero or greater is assigned to each mission vehicle. Values greater than 1 are used to distort the cost surface in favor of navigation support for that vehicle. This is based on the idea that each member in the network is "pulling on the same rope" and it is possible to optimize multilateration for a subset of vehicles by disproportionately penalizing any movement which adversely affects an important member with a large weight. For example, consider a 10-vehicle surveillance network consisting of 3 mission vehicles $\{1, 2, 3\}$ and 7 navigation support vehicles $\{4, 5, 6, 7, 8, 9, 10\}$. If the designer wishes to provide 0, 1, and 10-weighted autonomous navigation support for mission vehicles $\{1, 2, 3\}$, respectively, (4.9) could be devised.

$$\underline{\underline{W}} = \begin{bmatrix} 0 & & & & & & & & & \\ & 1 & & & & 0 & & & & \\ & & 10 & & & & & & & \\ & & & 1 & & & & & & \\ & & & & 1 & & & & & \\ & & & & & 1 & & & & \\ & & & & & & 1 & & & \\ & & & & & & & 1 & & \\ & & & 0 & & & & & 1 & \\ & & & & & & & & & 1 \end{bmatrix} \tag{4.9}$$

The weighted pseudolite cofactor matrix $\underline{\underline{\Sigma}}_c$ for this example can be computed using a least squares method shown in (4.10).

$$\underline{\underline{\Sigma}}_c = (\underline{\underline{P}}^T \underline{\underline{W}} \underline{\underline{P}})^{-1} \tag{4.10}$$

The definition of $\underline{\underline{\Sigma}}_c$ in (4.10) elucidates the importance of diverse angular coverage for reduction in relative positioning error variance. The shape of a cost surface defined by the $tr(\underline{\underline{\Sigma}}_c)$ for a practical network of pseudolites (not all vehicles collocated) is convex. Therefore, minimization of (4.11) is expected to be stable for all practical pseudolite networks. The cost function employed in the SCANS algorithm relies on minimization of the cofactor matrix dimensions with an anti-collision function:

$$SCANS_{Cost} = tr(\underline{\underline{\Sigma}}_c) + A^{\sum_1^N r_n^{-1}}, \tag{4.11}$$

where A is a desired anti-collision avoidance factor, N is the total number of nearby vehicles, and r_n is a noisy range measurement from vehicle n.

The value of the avoidance term in the cost function is approximately 1 until a vehicle travels near another vehicle. A determines the gradient of this anti-collision function. In this simulation, A was set to mitigate incursions inside 100 meters. Avoidance parameter tuning was accomplished iteratively via simulation. The SCANS algorithm is performed recursively by each navigation support vehicle with using the same weighting matrix.

4.3.4 Simulation.

All vehicles are spawned with a random uniform position on a 2000 m^2 2-D grid (Fig. 4.2). The navigation support vehicles use noisy ranging described in the previous

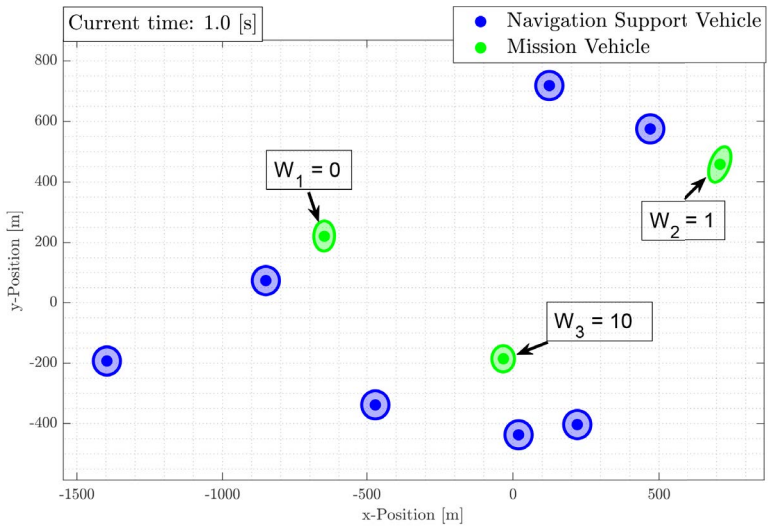

Figure 4.2: Snapshot for 10 Vehicle Network at t_i = 1.0 sec, 3 Mission Vehicles with weights of {0, 1, 10} and 7 Navigation Support Vehicles

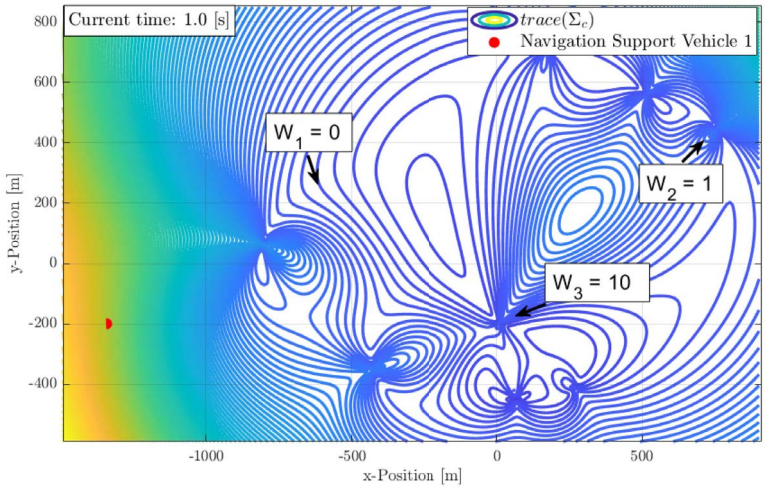

Figure 4.3: Weighted Cofactor Matrix Trace Surface Snapshot for 10 Vehicle Network at t_i = 1.0 sec, 3 Mission Vehicles with weights of {0, 1, 10} and 7 Navigation Support Vehicles from Perspective of Navigation Vehicle 1

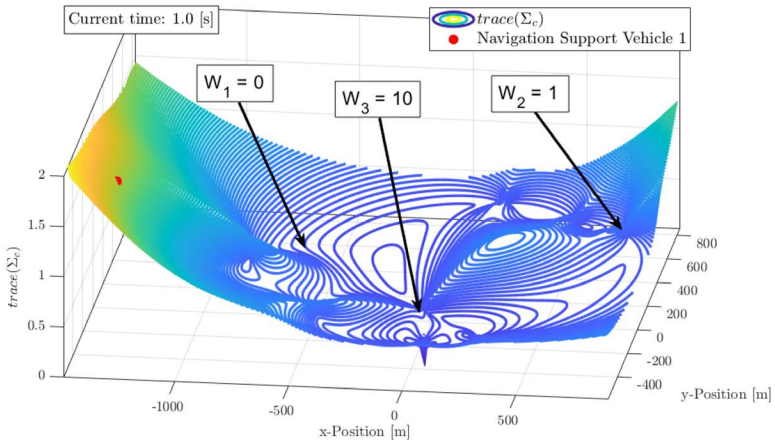

Figure 4.4: 3-D Weighted Cofactor Matrix Trace Surface Snapshot for 10 Vehicle Network at t_i = 1.0 sec, 3 Mission Vehicles with weights of {0, 1, 10} and 7 Navigation Support Vehicles from Perspective of Navigation Vehicle 1

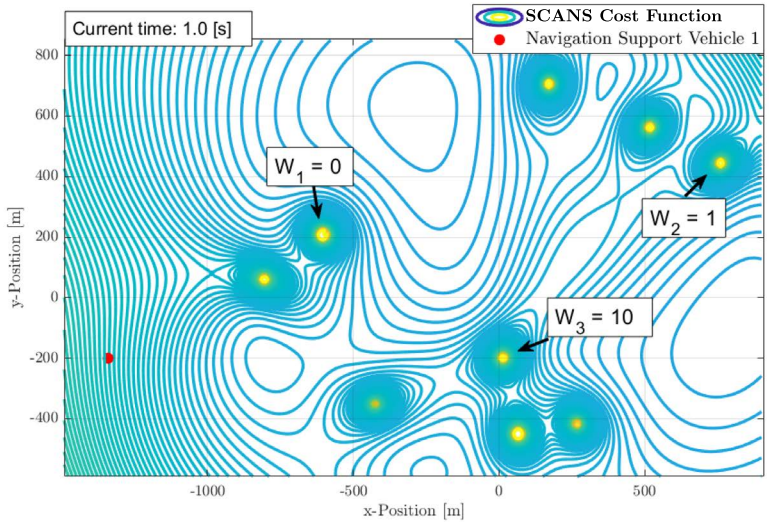

Figure 4.5: Weighted Cost Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 1 sec

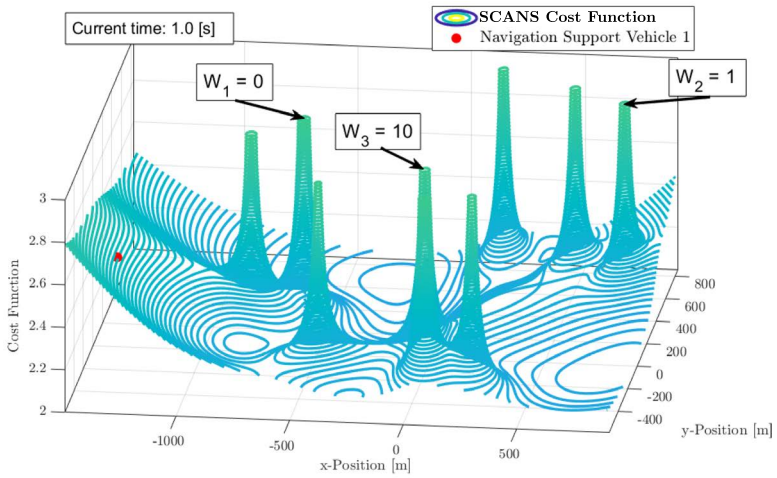

Figure 4.6: 3-D Weighted Cost Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 1 sec

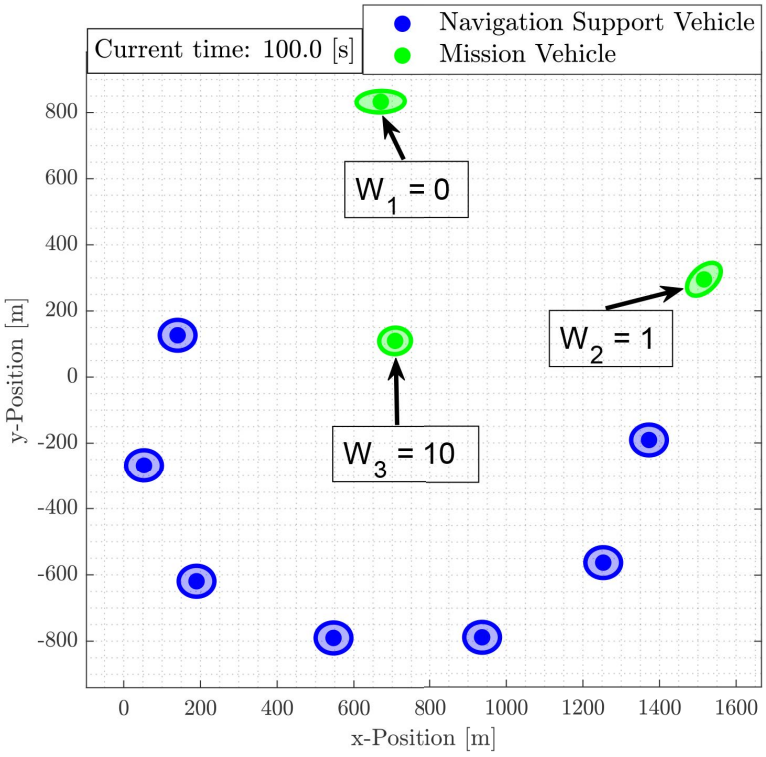

Figure 4.7: Snapshot for Stabilized 10 Vehicle Network at t_i = 100 sec, 3 Mission Vehicles and 7 Navigation Support Vehicles

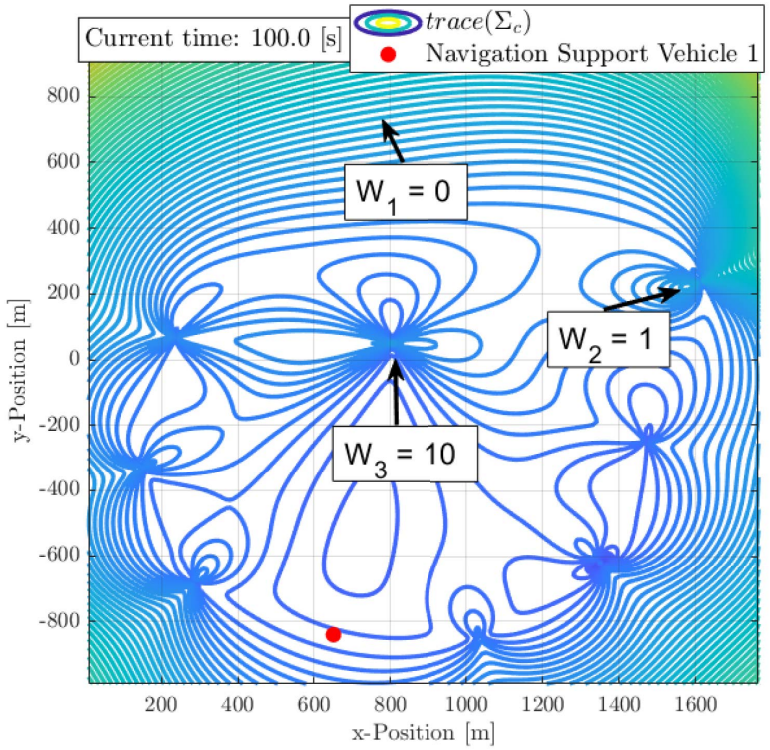

Figure 4.8: Weighted Cofactor Matrix Trace Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 100 sec

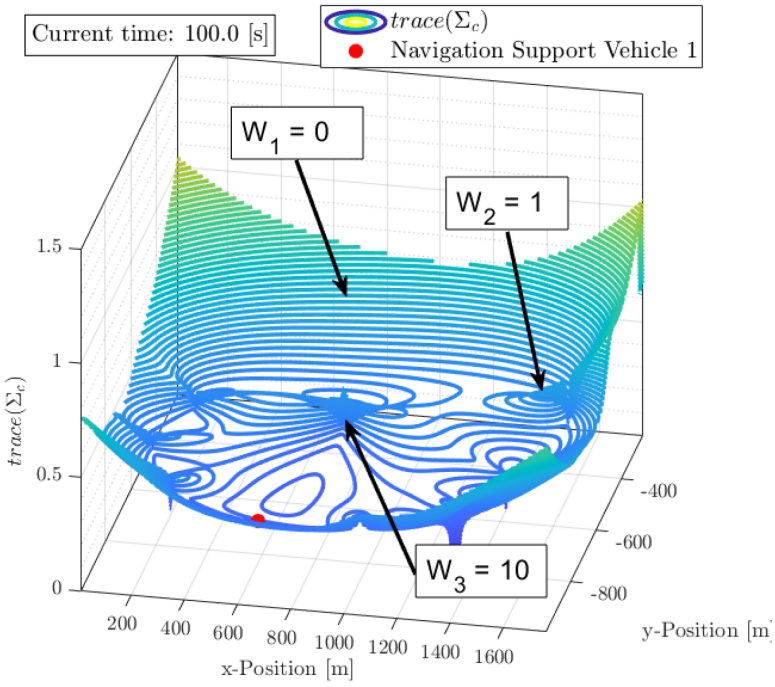

Figure 4.9: 3-D Weighted Cofactor Matrix Trace Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 100 sec

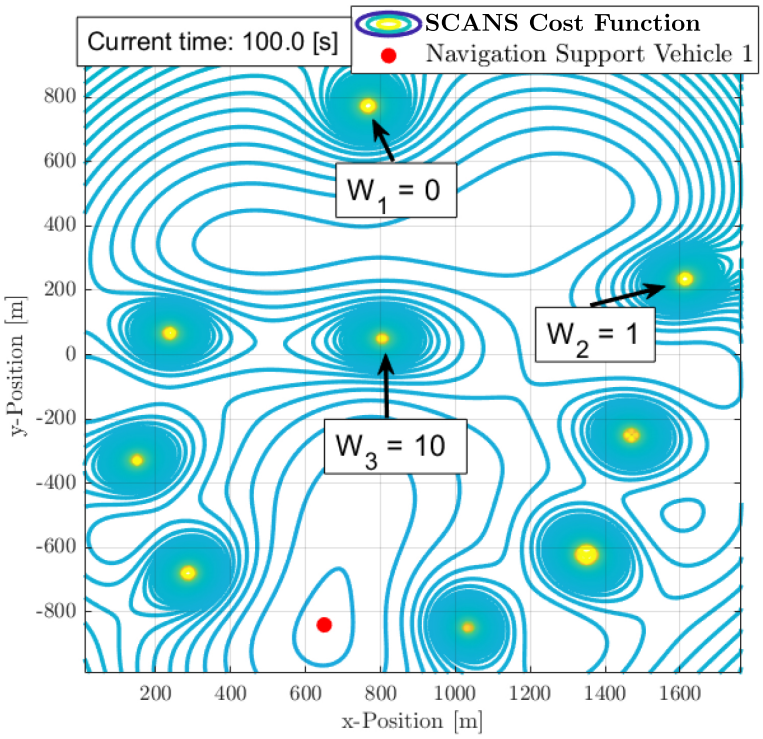

Figure 4.10: Weighted Cost Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 100 sec

section to simulate radio ranging to all vehicles in the surveillance network. Localization estimation is performed instantaneously without recursive Bayesian estimation. At each time step, $\Delta t = 1$ sec, mission vehicles instantaneously localize via multilateration by combining AWGN-corrupted navigation support vehicle positions ($\sigma_{mag} = 14.5$ m) over the surveillance network (4.2) with AWGN-corrupted radio ranging ($\sigma_{range} = 5$ m). Each simulation runs for $T = 150$ seconds. Mission vehicle position residuals are recorded for each time step and averaged over each run. The trajectories of the mission aircraft are random and unknown.

The navigation support vehicles are infinitely maneuverable with a maximum velocity of 25 m/s. The trajectory of each navigation vehicle is based on a particle swarm of size 10 inside a local circular area of radius 25 m centered on the location of the vehicle. Each navigation support vehicle trajectory is established by following the local gradient of the weighted cost function (4.11) at each time step. Once a local minimum is captured inside the search area, the vehicle follows the surface gradient to the minimum and maneuvers as necessary to maintain the local minimum. The vehicle will continue to follow the surface gradient towards a local minimum, resulting in multilateration optimization.

To simulate a random formation, mission aircraft are initialized with a stable 2-D velocity $V_0 = 10$ m/s initially in the $+X$ direction with a Gaussian random angular perturbation defined by (4.12).

$$\Theta_i = 0.8\Theta_{i-1} + 0.2\phi, \tag{4.12}$$

where time step size is 1 second, Θ is the current 2-D velocity angle and ϕ is a Normally distributed random angle between $[0, 2\pi]$ radians generated for each mission vehicle every time step t_i.

For example, Fig. 4.2 shows the results of initializing a 10-vehicle network with 3 mission vehicles with various weights. The weighting matrix $\underline{\underline{W}}$ is configured according

to (4.9). At $t = 1$ sec, the cost surface from the perspective of a random navigation support vehicle indicates that specific vehicle control is in the [+x] direction (bottom-left corner, Fig. 4.5). For the same instant, the $tr(\underline{\underline{\Sigma}}_c)$ surface from the perspective of the same navigation support vehicle is shown in Fig. 4.3. As expected, the random pseudolite network results in a globally convex surface (Fig. 4.4). The SCANS cost surface retains the globally convex shape which provides stable gradient-based cost minimization (Fig. 4.6). Note how mission vehicle 1 with $w_{11} = 0$ has no effect on the $tr(\underline{\underline{\Sigma}}_c)$ surface (Fig. 4.3), but causes a disturbance in the SCANS cost surface (Fig. 4.5). This is expected because mission vehicle 1 contributes to the avoidance parameter in the cost function. In other words, guidance includes avoidance but no multilateration consideration for a mission vehicle with a weight of 0.

The simulation is allowed to run for $t = 100$ sec until the autonomous swarm of 7 navigation vehicles has stabilized relative to the 3 mission vehicles (Fig. 4.7). The cost surface defined by (4.11) is visible from the perspective of the same navigation support vehicle. At $t = 100$ sec, the $tr(\underline{\underline{\Sigma}}_c)$ surface for the same navigation support vehicle is shown in Fig. 4.10. With the exception of zero-weighted mission vehicle 1, the peaks resulting from the anti-collision term visible in Fig. 4.10 coincide with the $tr(\underline{\underline{\Sigma}}_c)$ surface peaks in Fig. 4.8. As expected, the surface remains convex with a large optimized multilateration support region nearly centered around the heavily weighted mission vehicle 3 with $w_{33} = 10$ (See Fig. 4.9). The SCANS cost surface retains the globally convex shape with a stable optimized region resulting from distributed greedy minimization (See Fig. 4.11). Similarly, navigation support stabilized at $t = 100$ sec for mission vehicle 2 at an intermediate location on the cost surface, corresponding to unity weighting. Lastly, mission vehicle 1 is given no navigation support consideration and exists in a random non-optimal location (Fig. 4.7).

The following pseudocode describes how SCANS is implemented on each navigation support vehicle.

SCANS for Weighted Distributed Navigation Support

Performed by each Autonomous Navigation Support Vehicle

Initialization Time $t = 0$ sec, Uniform Random Spawn All Vehicles across $2000m^2$ 2-D grid
Initialize Particle Swarm ▶ 10 particles, kinematic search area, radius = 25m, optimizedPosition = $[x_n(t), y_n(t)]$
Initialize Avoidance Parameter A
Initialize Design Matrix $\underline{\underline{P}}$
Initialize Shared Weight Matrix $\underline{\underline{W}}$
Initialize Convergence Threshold $threshold = 0.01$
while $t < t_{max}$ **do**
    Input: Noise-Corrupted Vehicle Positions $\{x_1(t), y_1(t), ..., x_N(t), y_N(t)\}$
    Input: Noise-Corrupted Ranges $\{r_1(t), ...r_N(t)\}$
    Populate Design Matrix $\underline{\underline{P}}$
    Construct Pseudolite Cofactor Matrix, $\underline{\underline{\Sigma}}_c = (\underline{\underline{P}}^T \underline{\underline{W}} \underline{\underline{P}})^{-1}$ (4.10)
    while $\Delta optPosition > threshold$ **do**
        Minimize SCANS Cost Function, $SCANS_{Cost} = tr(\underline{\underline{\Sigma}}_c) + A^{\Sigma_1^N r_n^{-1}}$, (4.11)
    end while
    $t = t + 1$
    $optimizedPosition = [x_n(t), y_n(t)]$
    Output: $optimizedPosition$
end while

4.4 Numerical Results

Four 10-vehicle surveillance network configurations (7-navigation support vehicles, 3 mission vehicles) were simulated with various mission vehicle importance weights to assess the impacts on mission vehicle RMSE performance. The mean position error for each vehicle over each run is recorded and used to produce a 10K trial 2D RMSE distribution. In total, more than 1600 hours of flight time were simulated. In each set of 10K trials, the 7 Navigation Support Vehicle RMSE distributions converged to the distribution visible in Fig. 4.12. A grand mean RMSE was calculated for mission vehicle and is visible in Table 1. Figs. 4.13-4.16 contain violin plot comparisons of the 2-D position RMSE distributions for Monte Carlo Trials 1-4, respectively.

Based on simulation results, multilateration geometry drives the shape of the position RMSE distribution. We expect a positive-shift in mean and less Gaussian behavior (i.e. longer positive tails) as multilateration geometry to anchors worsens. The quality of navigation support (i.e. multilateration geometry) is determined by the weight of the mission vehicle relative to the total weight of the swarm. Trial 4 (Fig. 4.16) shows a 36% mean reduction in RMSE with a distribution that approaches Normality for SCANS-weighted formation versus non-optimized mission vehicles in the formation. The non-optimized vehicles exhibit highly positive-skewed error distributions which are suboptimal for use in recursive estimators that make use of white Gaussian noise assumptions regardless of the actual error distributions.

4.5 Large Swarm Behavior

Implementation of large SCANS-based swarm implementations are straightforward due to the distributed, greedy architecture. In fact, any swarm size can be simulated with

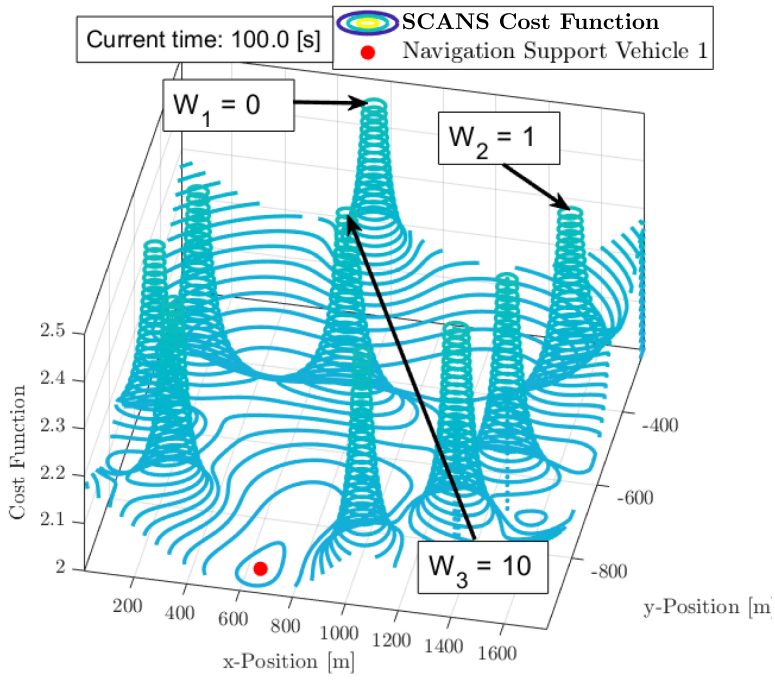

Figure 4.11: 3-D Weighted Cost Surface for 10 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 100 sec

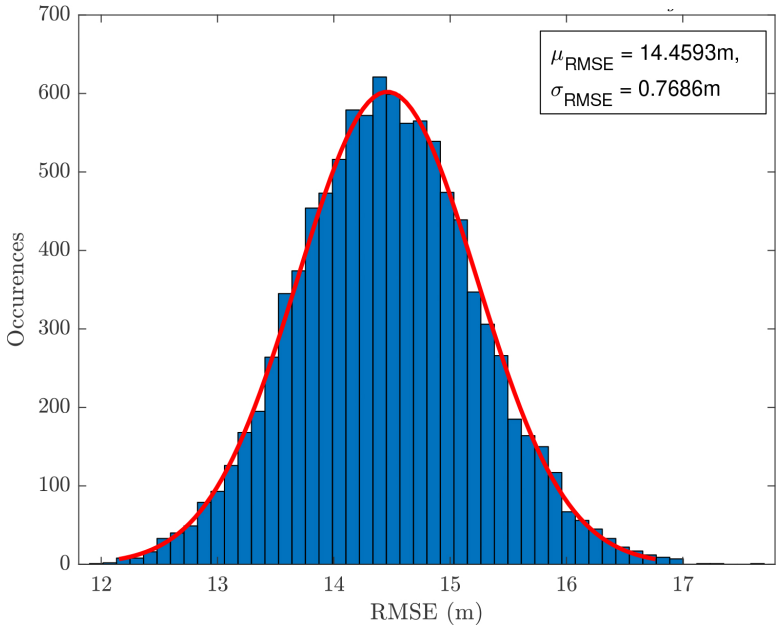

Figure 4.12: Example 2-D Position RMSE for Magnetic Navigation Support Vehicle 4, Trial 3

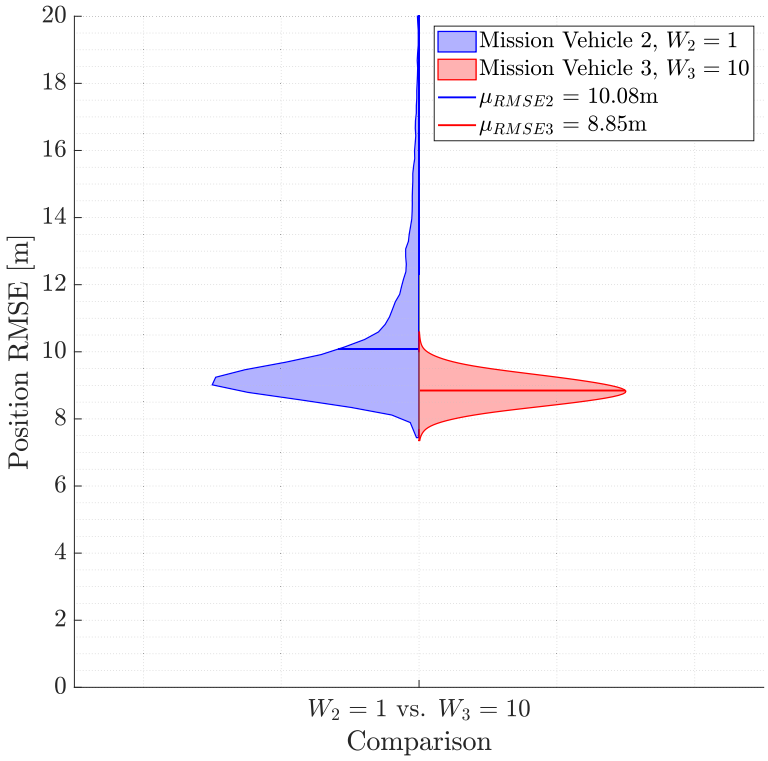

Figure 4.13: 2-D Position RMSE Mission Vehicle 2 vs Mission Vehicle 3, Monte Carlo Trial 1

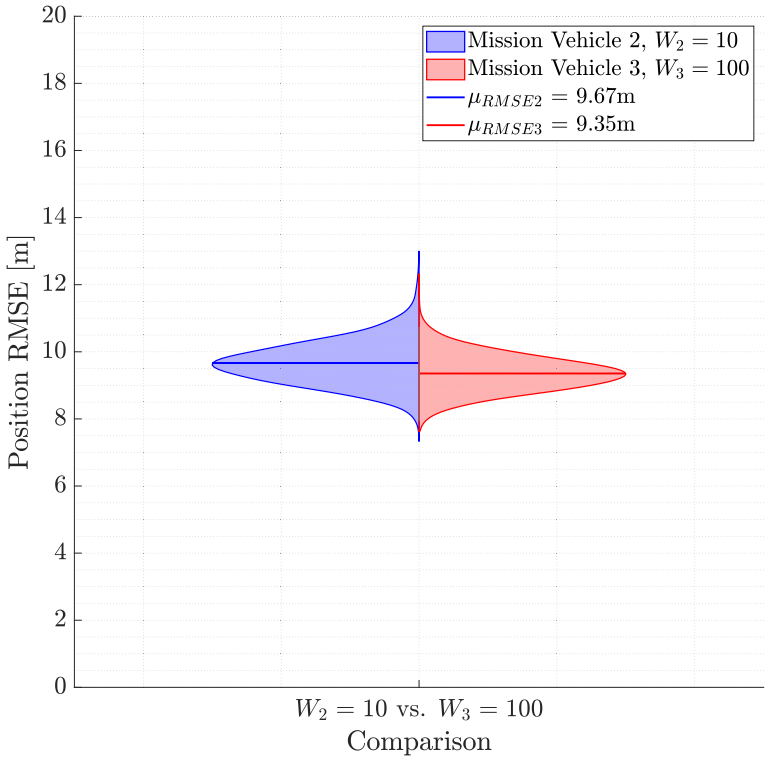

Figure 4.14: 2-D Position RMSE Mission Vehicle 2 vs Mission Vehicle 3, Monte Carlo Trial 2

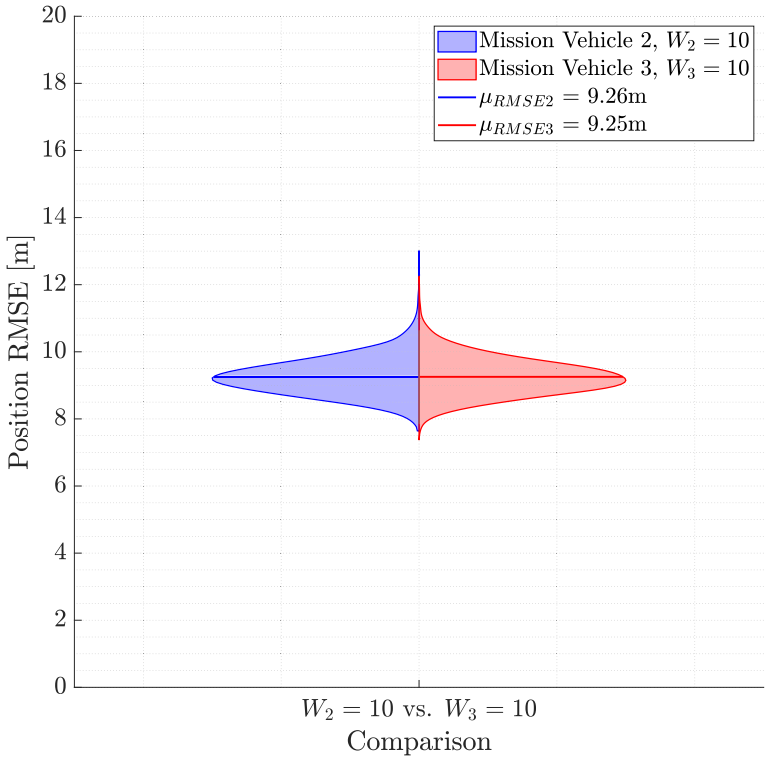

Figure 4.15: 2-D Position RMSE Mission Vehicle 2 vs Mission Vehicle 3, Monte Carlo Trial 3

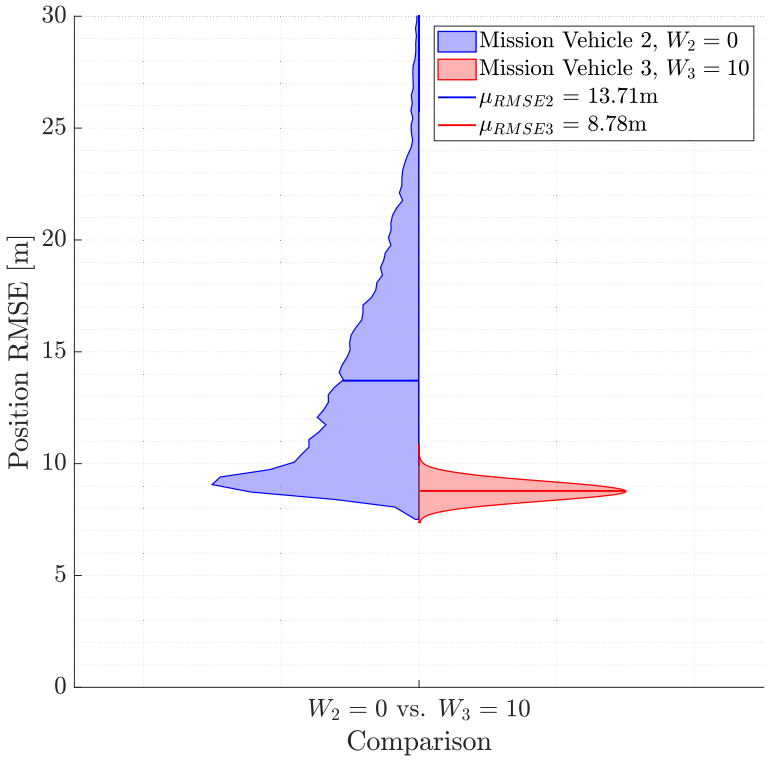

Figure 4.16: 2-D Position RMSE Mission Vehicle 2 vs Mission Vehicle 3, Monte Carlo Trial 4

Mission Vehicle Weights, W_1, W_2, W_3	Mission Vehicle 1 μ_{RMSE} (median)	Mission Vehicle 2 μ_{RMSE} (median)	Mission Vehicle 3 μ_{RMSE} (median)
0, 1, 10	11.54 m (10.59 m)	10.08 m (9.42 m)	8.85 m (8.84 m)
0, 10, 100	9.99 m (9.43 m)	9.67 m (9.64 m)	9.35 m (9.34 m)
0, 10, 10	10.28 m (9.49 m)	9.26 m (9.23 m)	9.25 m (9.22 m)
0, 0, 10	13.73 m (12.61 m)	13.71 m (12.56 m)	8.78 m (8.78 m)

Table 4.1: Grand Mean (Median) 2-D Position RMSE for 10-Vehicle Network with Varying Mission Vehicle Weights, (4) 10K Monte Carlo Trials

sufficient available processing cycles. It should be noted that the accuracy of multilateration support increases with the size of the navigation support swarm. In other words, relative navigation support vehicle geometry is less important as the size of the swarm increases as a result of increased Fisher information available to each mission vehicle. In a 50-vehicle simulation, initialized at $t = 0$ s, it is clear that the convex nature of the SCANS cost surface is retained resulting in stable and convergent guidance for navigation support vehicles. Fig. 4.17 shows the results of initializing a 50-vehicle network with 47 navigation support vehicles and 3 mission vehicles with weights of 0, 5, and 50, respectively. Fig. 4.18 shows the $tr(\underline{\underline{\Sigma}}_c)$ surface for the same network configuration. This surface is globally convex (Fig. 4.19). Fig. 4.20 shows a contour plot of the SCANS cost surface. As expected, the cost

surface retains the globally convex shape (Fig. 4.21) with local maxima resulting from the avoidance parameter in (4.11).

Fig. 4.22 shows the results of a stabilized 50-vehicle network with 47 navigation support vehicles and 3 mission vehicles with weights of 0, 5, and 50, respectively at $t_i = 250$ sec. Fig. 4.23 shows the $tr(\underline{\underline{\Sigma}}_c)$ surface for the same network configuration. This surface shows a large optimized CRLB region containing both the weighted mission vehicles (Fig. 4.24). Fig. 4.25 shows the stabilized SCANS cost surface. As expected, the stabilized cost surface remains convex overall with a stable, predictable swarm formation based on greedy local cost minimization.

4.6 Chapter Summary and Future Work

This chapter presented a stable, distributed simulation environment with a pluggable cost function and demonstrated its effectiveness for consensus-like multilateration optimization. As long as a globally convex cost surface can be generated, these simulation techniques can be applied to solve very complex distributed mathematical problems in real-time. In this example, we have shown that this enables vehicles equipped with all-source capability to provide localization improvement for other non-cooperative mission vehicles. As the weight assigned to the mission vehicles is increased relative to the total weight of the swarm, mission vehicle position RMSE is reduced and approaches a Normal-Gaussian distribution. This effort provides real-time distributed vehicle control for consensus-like offboard multilateration optimization in a simulated time-synchronized surveillance network. Future applications with this simulation technique include control schemes to optimize magnetic navigation accuracy using scalar map gradients.

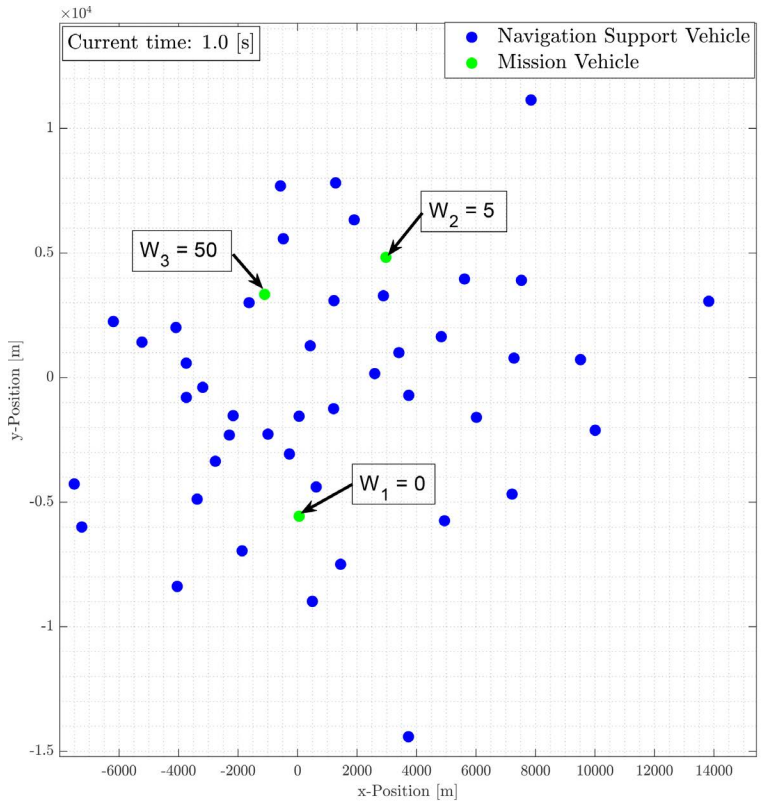

Figure 4.17: Snapshot for Initialized 50 Vehicle Network at t_i = 1 sec, 3 Mission Vehicles and 47 Navigation Support Vehicles

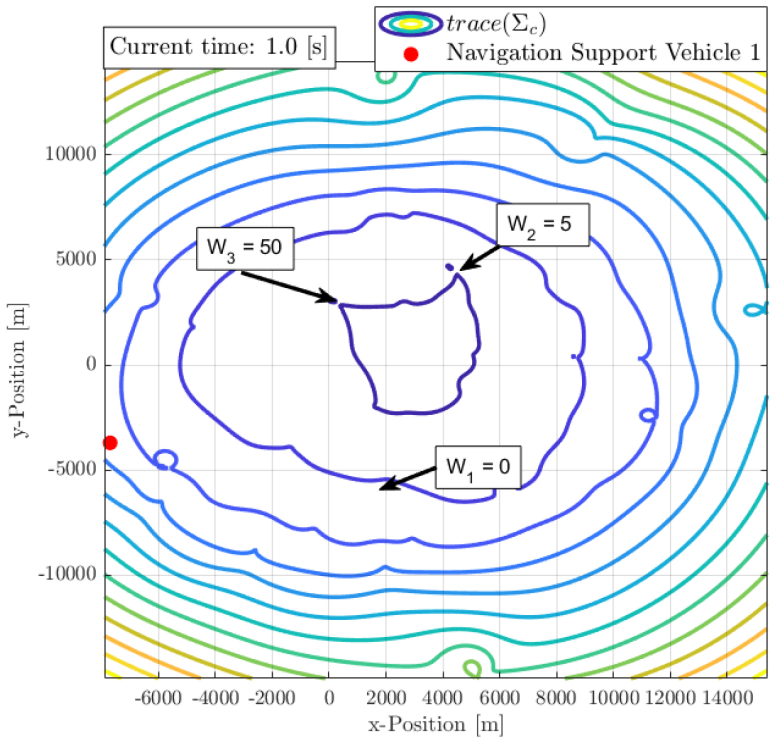

Figure 4.18: Weighted Cofactor Matrix Trace Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 1 sec

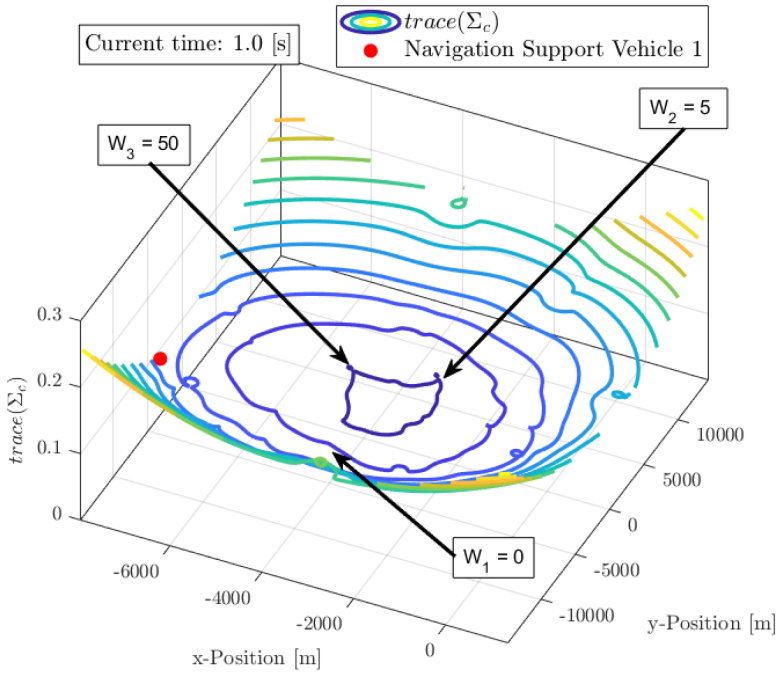

Figure 4.19: Weighted Cofactor Matrix Trace Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 1 sec

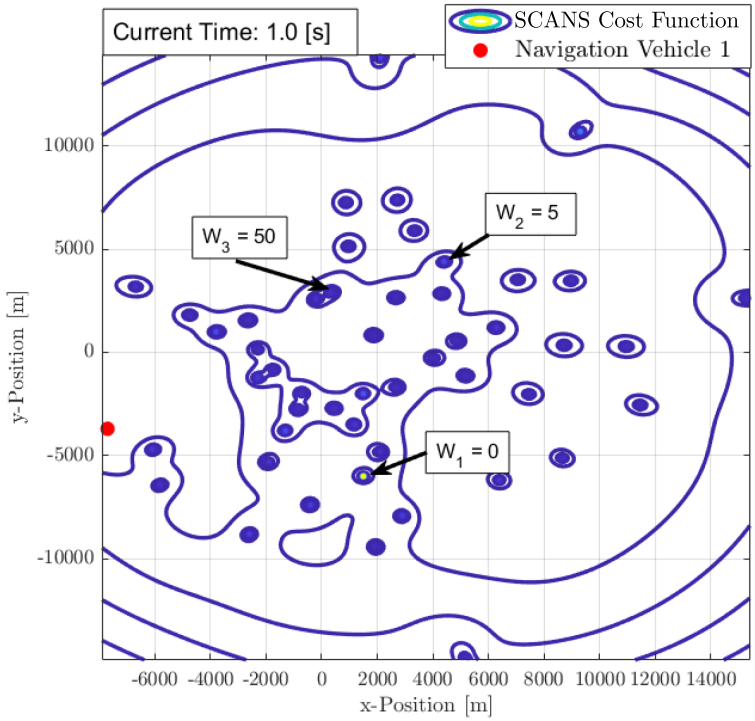

Figure 4.20: Weighted Cost Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at $t_i = 1$ sec

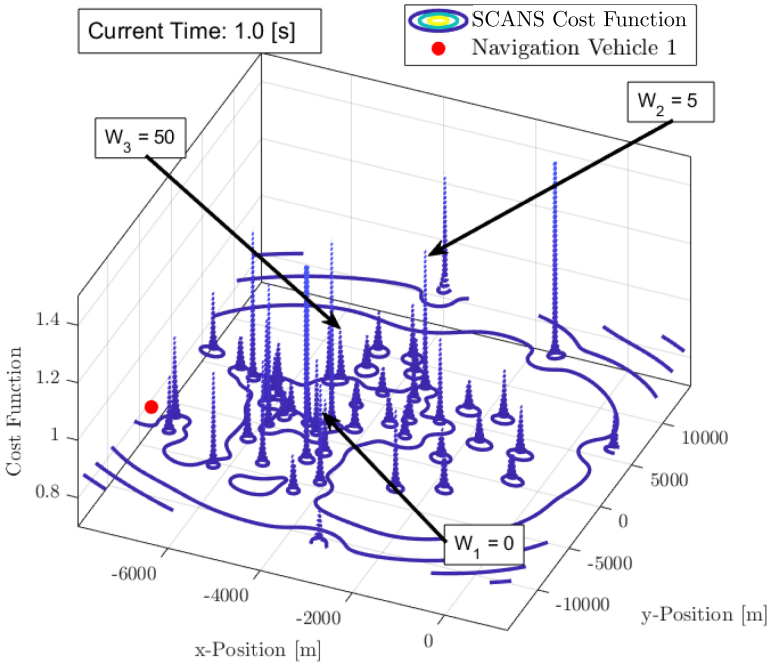

Figure 4.21: 3-D Weighted Cost Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 1 sec

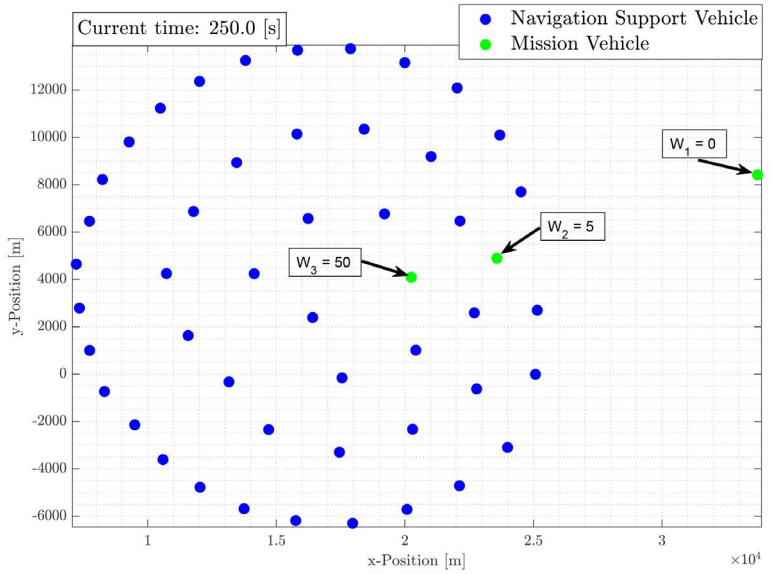

Figure 4.22: Snapshot for Stabilized 50 Vehicle Network at t_i = 250 sec, 3 Mission Vehicles and 47 Navigation Support Vehicles

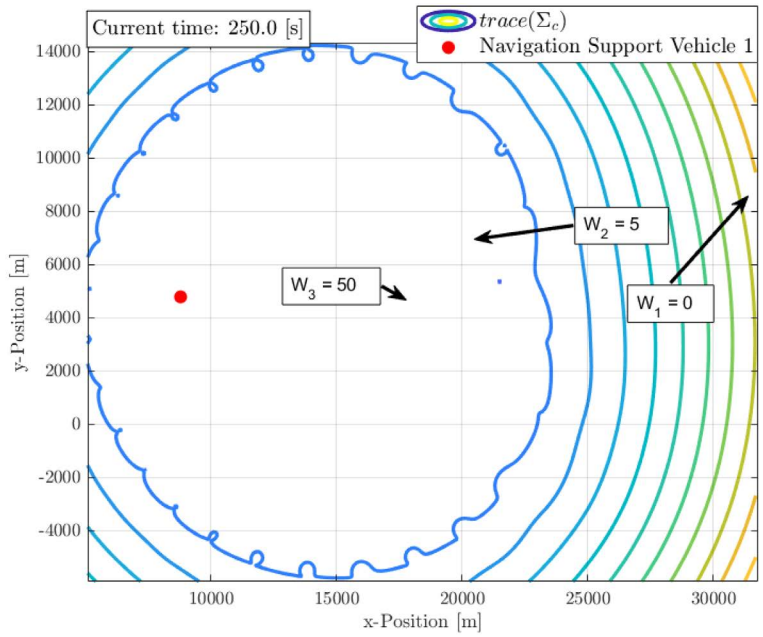

Figure 4.23: Weighted Cofactor Matrix Trace Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 250 sec

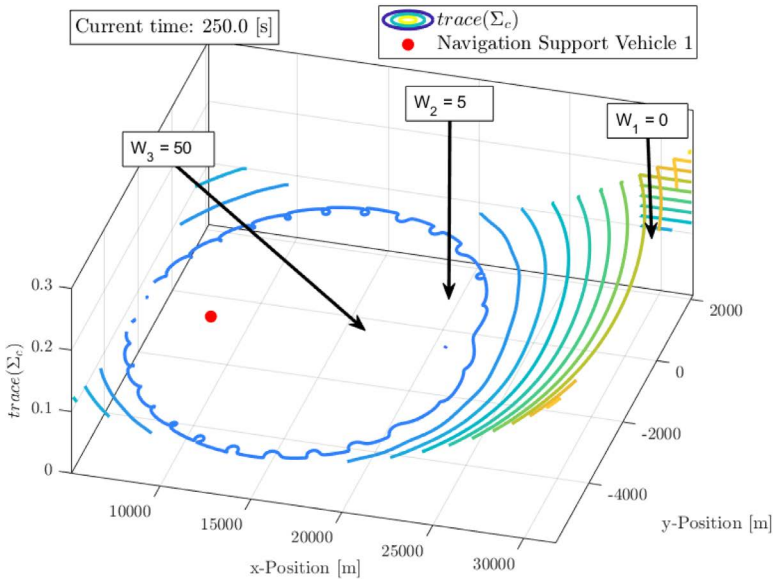

Figure 4.24: 3-D Weighted Cofactor Matrix Trace Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 250 sec

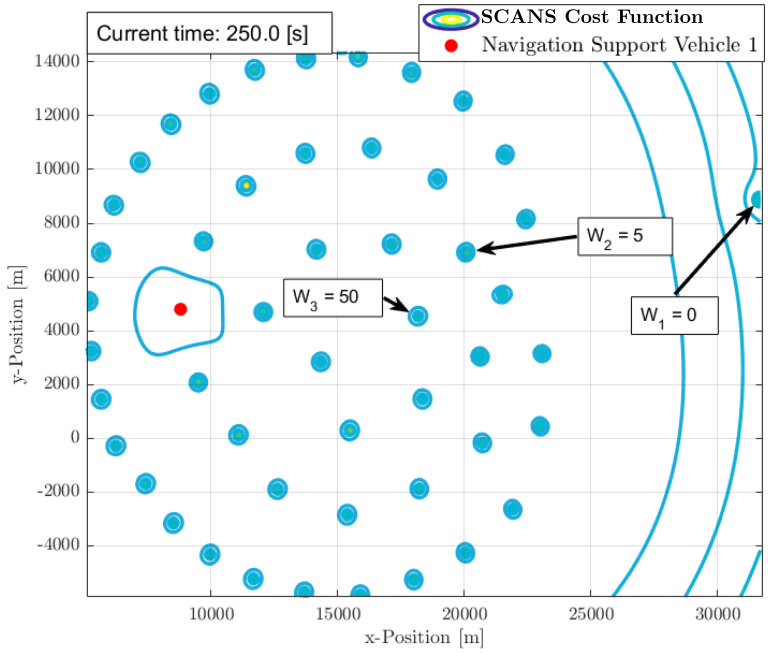

Figure 4.25: Weighted Cost Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 250 sec

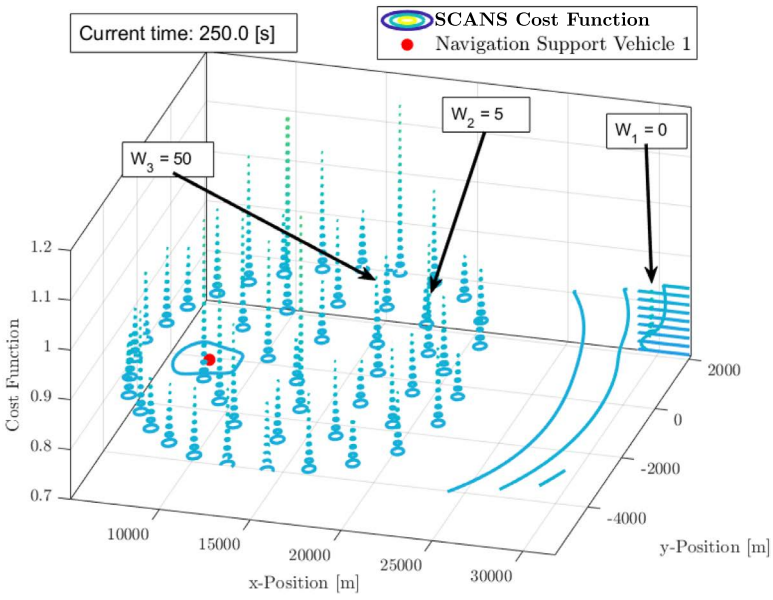

Figure 4.26: 3-D Weighted Cost Surface for 50 Vehicle Network from Perspective of Navigation Support Vehicle 1 at t_i = 250 sec

V. Stable Observability Monitoring

The Autonomous and Resilient Management of All-source Sensors (ARMAS) framework monitors residual-space test statistics across unique sensor-exclusion banks of filters, (known as *subfilters*) to provide a resilient, fault-resistant all-source navigation architecture with assurance. A critical assumption of this architecture, demonstrated in this chapter, is fully overlapping state observability across all subfilters. All-source sensors, particularly those that only provide partial state information (altimeters, TDoA, AOB, etc.) do not intrinsically meet this requirement.

This chapter presents a novel method to monitor real-time overlapping position state observability and introduces an "observability bank" within the ARMAS framework, known as Stable Observability Monitoring (SOM). SOM uses real-time stability analysis to provide an intrinsic awareness to ARMAS of the capabilities of the fault detection and exclusion (FDE) functionality. We define the ability to maintain consistent all-source FDE to recover failed sensors as *navigation resilience*. A resilient FDE capability then is one that is "aware" of when it requires more sensor information to protect the consistency of the FDE and integrity functions from corruption. SOM is the first demonstration of such a system, for all-source sensors, that the authors are aware.

A multi-agent 3-D environment simulating both GNSS and position and velocity alternative navigation sensors was created and individual GNSS pseudorange sensor anomalies are utilized to demonstrate the capabilities of the novel algorithm. This chapter demonstrates that SOM seamlessly integrates within the ARMAS framework, provides timely prompts to augment with new sensor information from other agents, and indicates when framework stability and preservation of all-source navigation integrity are achieved.

5.1 Introduction

Introduced in 2018, ARMAS provided a generalized framework for real-time management of heterogeneous, asynchronous all-source sensors [43]. This framework was resilient to corruption from mismodeled, uncalibrated, and faulty sensors and was accomplished by combining sensor validation, FDE, recalibration, and remodeling modes in a single architecture. Sensor-Agnostic All-source Residual Monitoring (SAARM) was designed to provide all-source Fault Detection and Exclusion (FDE) and navigation integrity functions within the ARMAS framework [44]. In this context, all-source navigation resiliency is the ability to maintain consistent all-source FDE operations with the ability to recover failed sensors. The pluggable Bayesian filters provided by the SCORPION estimation architecture afforded needed flexibility to spawn, propagate, and remove estimation filters on the fly [47]. SAARM required the designer to maintain a set of unique navigation subfilters (each unique subfilter excludes measurements from a different sensor) to maintain resilience to a single simultaneous sensor failure. A primary assumption of the FDE and integrity functions was the ability to maintain overlapping position state observability. This chapter presents a novel method to monitor real-time overlapping position state observability and introduces an "observability bank" within the ARMAS framework. These additions to ARMAS use real-time observability analysis at the layer 2 subfilter level (each unique layer 2 subfilter excludes measurements from two different sensors) to provide a timely indication to augment the framework with new sensor data, thus protecting the consistency of the ARMAS FDE and integrity functions from corruption.

5.2 Background

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

The ARMAS framework was designed to gracefully recover from multiple types of failure modes (bias, model mismatched, and/or sensor miscalibration) while attempting

to maintain a consistent, uncorrupted navigation estimate. ARMAS employs a set of SCORPION pluggable EKF estimators to address the following nonlinear navigation problem:

$$\dot{\mathbf{x}}(t) = \boldsymbol{f}[\mathbf{x}(t), \boldsymbol{\alpha}(t), \boldsymbol{u}(t), t] + \boldsymbol{G}(t)\boldsymbol{w}(t) \tag{5.1}$$

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

$$\mathbf{z}_k^{[j]} = \mathbf{h}^{[j]}[\mathbf{x}(t), \alpha^{[j]}(t), \mathbf{u}(t), t, \mathbf{p}^{[j]}] + \boldsymbol{\nu}_k^{[j]}, \tag{5.2}$$

where $\mathbf{h}^{[j]}$ is the nonlinear measurement function for the j^{th} sensor, $\alpha^{[\mathbf{j}]}$ is an $L \times 1$ subset of α 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

$$\mathbf{r}_k^{[j]} = \mathbf{z}_k^{[j]} - \mathbf{h}^{[j]}\left[\hat{\mathbf{x}}_k^-, \hat{\alpha}_k^{[j]-}, \mathbf{u}_k, t_k, \hat{\mathbf{p}}_k^{[j]}\right], \tag{5.3}$$

where $\hat{\mathbf{x}}_k^-$, $\hat{\alpha}_k^{[j]-}$, and $\hat{\mathbf{p}}_k^{[j]}$ are estimated quantities. Assuming white Gaussian noise, the measurement residual from (5.3) is expected to follow the distribution

$$\mathbf{r}_k^{[j]} \hookrightarrow \mathcal{N}\left(\mathbf{0}_{N\times 1}, \mathbf{S}_k^{[j]}\right), \tag{5.4}$$

$$\mathbf{S}_k^{[j]} = \mathbf{H}_k^{[j]}\mathbf{P}_k^-\mathbf{H}_k^{[j]^T} + \mathbf{R}_k^{[j]}, \tag{5.5}$$

where $\mathbf{P}_k^-$ is the $(N + M) \times (N + M)$ state estimate 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 [42]. In validation mode, ARMAS employs a likelihood function to monitor the statistical distribution of a user-defined monitoring period composed of recent Kalman pre-update residuals. A Chi-square, χ^*, test statistic is used to detect excursions outside a user-defined threshold across the sampling period. Sensors in validation mode are excluded from impacting the main state estimates using a Schmidt partial update [9]. Trusted sensors are directly 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 5.2.2.

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 $\alpha^{[j]}$ 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 $\alpha^{[j]}$), 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 (5.4) 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 [43]. Figure 5.1 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.

One assumption that is not explicitly discussed in [43] is that ARMAS requires overlapping state observability [31] to detect anomalous sensor behavior. 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 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. The criticality of this assumption for ARMAS is highlighted in the analysis below.

5.2.2 Sensor-Agnostic All-source Residual Monitoring (SAARM) .

SAARM assumes a system form of

$$\dot{\mathbf{x}}(t) = \boldsymbol{f}[\mathbf{x}(t), \boldsymbol{\alpha}(t), \boldsymbol{u}(t), t] + \boldsymbol{G}(t)\boldsymbol{w}(t) \tag{5.6}$$

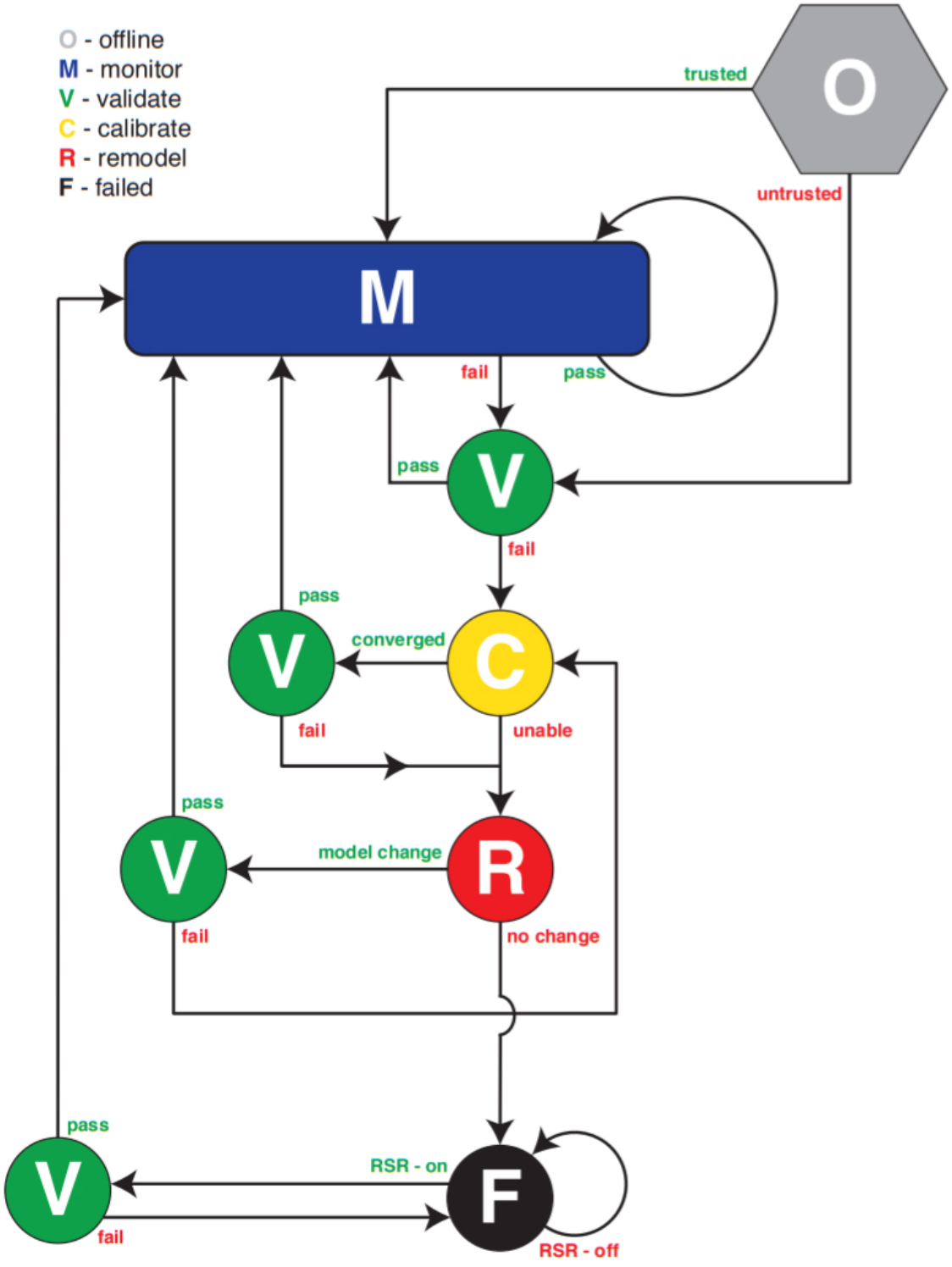

Figure 5.1: ARMAS Framework State Diagram [43]. A sensor begins at point **O** (origin) and is "trusted" or "untrusted". SAARM and the new contribution SOM reside within the **M** (Monitoring) mode.

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

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

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

$$\mathbf{z}^{[i]}(t_k) = \mathbf{h}^{[i]}\big[\hat{\mathbf{x}}^{[j]}(t_k^-), \mathbf{u}(t_k), t_k\big] \tag{5.7}$$

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 $Z \times 1$ defined by covariance matrix $\mathbf{R}^{[i]}(t_k)$. The pre-update measurement estimate for sensor i from filter j is defined by

$$\hat{\mathbf{z}}^{[i,j]}(t_k) = \mathbf{h}^{[i]}\big[\hat{\mathbf{x}}^{[j]}(t_k^-), \mathbf{u}(t_k), t_k\big], \tag{5.8}$$

where the estimated covariance matrix is defined by

$$\mathbf{P}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}{}^{[i,j]}(t_k^-) = \mathbf{H}^{[i]}(t_k^-)\mathbf{P}_{\hat{\mathbf{x}}\hat{\mathbf{x}}}{}^{[j]}(t_k^-)\mathbf{H}^{[i]}(t_k^-)^T. \tag{5.9}$$

Using (5.8) and (5.9), 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

$$\mathbf{r}^{[i,j]}(t_k) = \mathbf{z}^{[i]}(t_k) - \hat{\mathbf{z}}^{[i,j]}(t_k^-), \tag{5.10}$$

$$\mathbf{P}^{[i,j]}_{rr}(t_k) = \mathbf{R}^{[i]}(t_k) + \mathbf{P}^{[i,j]}_{\hat{z}\hat{z}}(t_k^-). \tag{5.11}$$

Fault detection relies on computing a moving average of recent residual-space test statistics formed by pre-update residual vectors from (5.10) and (5.11). ARMAS is designed to detect any sensor behavior which is inconsistent with the stated measurement model within the limitations of the stated significance level, $alpha$. Although not examined in this chapter specifically, the ARMAS framework provides additional options for the user to provide candidate models and/or calibration schemes which could be used to validate and recover

a failed sensor. The likelihood function focuses on a single residual-space statistic derived from the Mahalanobis distance, d, given by

$$d^2 = (\mathbf{y} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{y} - \boldsymbol{\mu}), \tag{5.12}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean and covariance of a Z_i-dimensional Gaussian distribution. It is known that a sum of M independent d^2 distances follows a χ^* distribution with Z degrees of freedom [21] given by

$$\chi^* = \sum_{s=k}^{k+M} d^2(t_s), \tag{5.13}$$

$$d^2(t_k) = \mathbf{r}^T(t_k)[\mathbf{P}_{rr}(t_k)]^{-1}\mathbf{r}(t_k). \tag{5.14}$$

The set of pre-update residuals is known to be a zero-mean, Gaussian sequence [56]. The fault detection test for M pre-residuals is composed of the following hypotheses:

$$H_0 : \chi^*_{[i,j]} < \chi^2(1 - \alpha/2, M \times Z_i) \tag{5.15}$$

$$H_1 : \chi^*_{[i,j]} > \chi^2(1 - \alpha/2, M \times Z_i) \tag{5.16}$$

where α is the probablity of false alarm and M is the number of averaged pre-residual samples. 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 agreement of all other 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). 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* [43].

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

$$\mathbf{T}(i,j) = \begin{cases} 0, \text{ Sensor } i \text{ not associated with filter } j \\ 0, \ \chi^*_{[i,j]} < \chi^2(1-\alpha/2, M \times Z_i), \text{ No Fault, } H_0 \\ 1, \ \chi^*_{[i,j]} > \chi^2(1-\alpha/2, M \times Z_i), \text{ Fault, } H_1 \end{cases} \tag{5.17}$$

Figure 5.2 shows the relationship between I sensors and J subfilters required for "fault agreement" 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 covariance matrix, $\mathbf{R}^{[i]}$, from the i^{th} sensor. Each column contains the estimated measurement, $\hat{\mathbf{z}}^{[i,j]}$, and its 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 an agreement 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 5.1. Of note, SAARM is able to detect the occurrence of multiple simultaneous faults but is only able to provide multiple fault exclusion when initialized with additional subfilter layers.

In summary, SAARM provides all-source sensor FDE and integrity for various sensor fault types. 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 χ^*

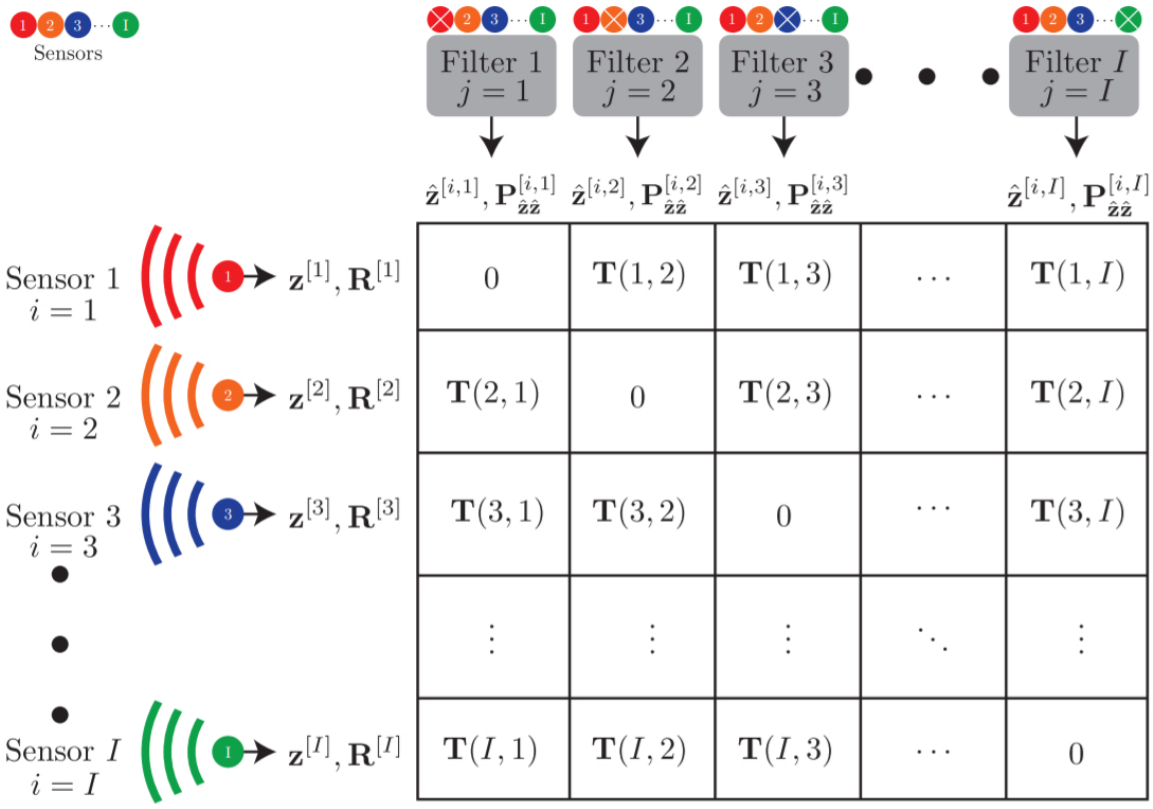

Figure 5.2: SAARM T-Matrix for $i = 1...I$ All-Source Sensors [44]. If Sensor 3 were faulty, for example, each subfilter that includes that sensor would report a fault (all across row three) and only subfilter 3 would remain consistent. Filter 3 would then be promoted to the main filter level and a new bank of FDE subfilters must be populated.

statistical tests of pre-update measurement residuals. Fault exclusion is based on a subfilter agreement approach which is tallied in a novel **T** matrix. If every subfilter has position state observability, then 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. The next section describes the motivation for a novel layer in ARMAS used for real-time observability analysis.

5.2.3 Motivation for Stable Observability Monitoring (SOM).

The ARMAS framework with SAARM was originally conceived and simulated with basic linear 2D position and velocity sensors and assumed fully overlapping state observability within the FDE layer. All-source sensors, particularly those that only provide partial state information do not intrinsically exhibit this characteristic. In collaborative navigation scenarios, retention of autonomy is desirable as long as a stable, resilient solution can be maintained. Another key motivation for Stable Observability Monitoring (SOM) is the ability to determine when to augment with collaborative information and a method to measure the sufficiency of the collaborative information. In early 2020, the ARMAS framework was applied to a flight test dataset [4] (Figure 5.3). The flight was conducted by the Autonomy and Navigation Technology (ANT) Center at the Air Force Institute of Technology (AFIT) on 12 October 2018 at Camp Atterbury, IN where a small Unmanned Aerial System (sUAS) took off and landed at Himsel Army Airfield (HAA). During the 27-minute dataset, the aircraft flew patterns at approximately 250 and 100 meters above the local surface. The aircraft used the ANT Center's Scorpion framework [47] to provide a GNSS/INS-coupled 'truth' navigation solution. This analysis consists of individual pseudorange measurements extracted from six Global Positioning System (GPS) Satellite Vehicles (SV) for nonlinear processing in ARMAS as individual sensors.

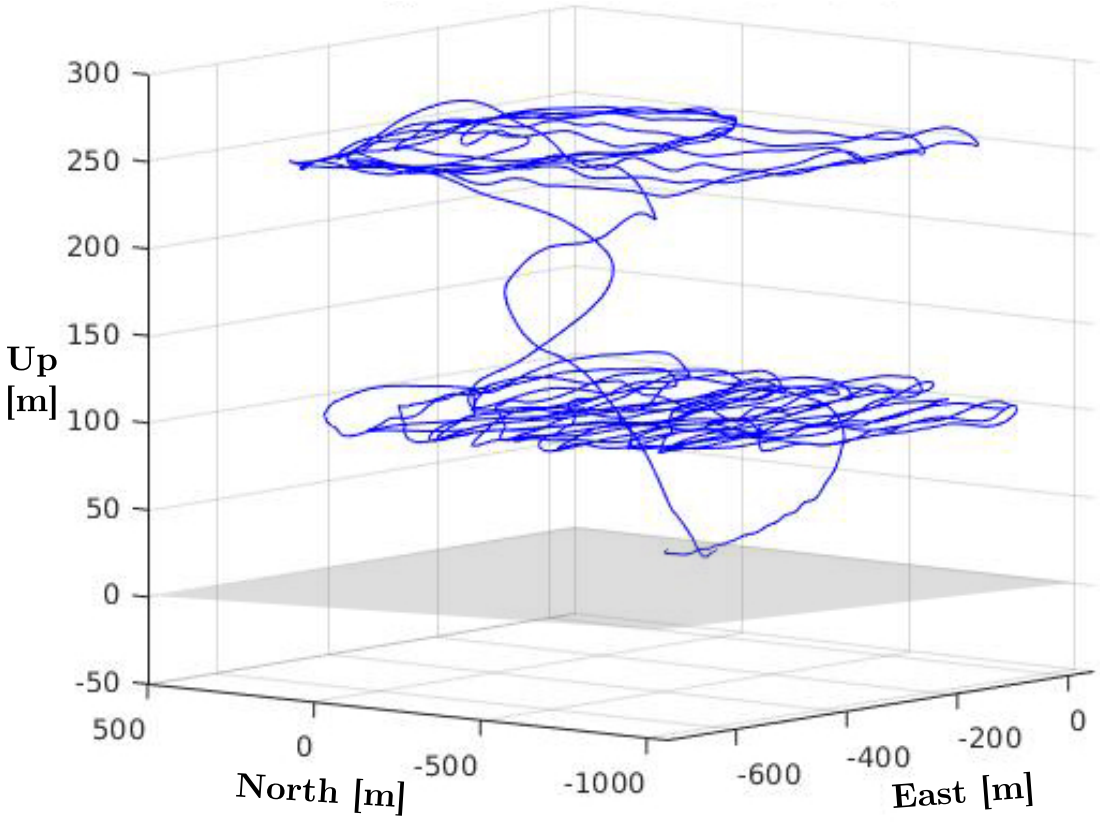

Figure 5.3: sUAS Flight Test Data, 12 Oct 2018, Camp Atterbury, IN

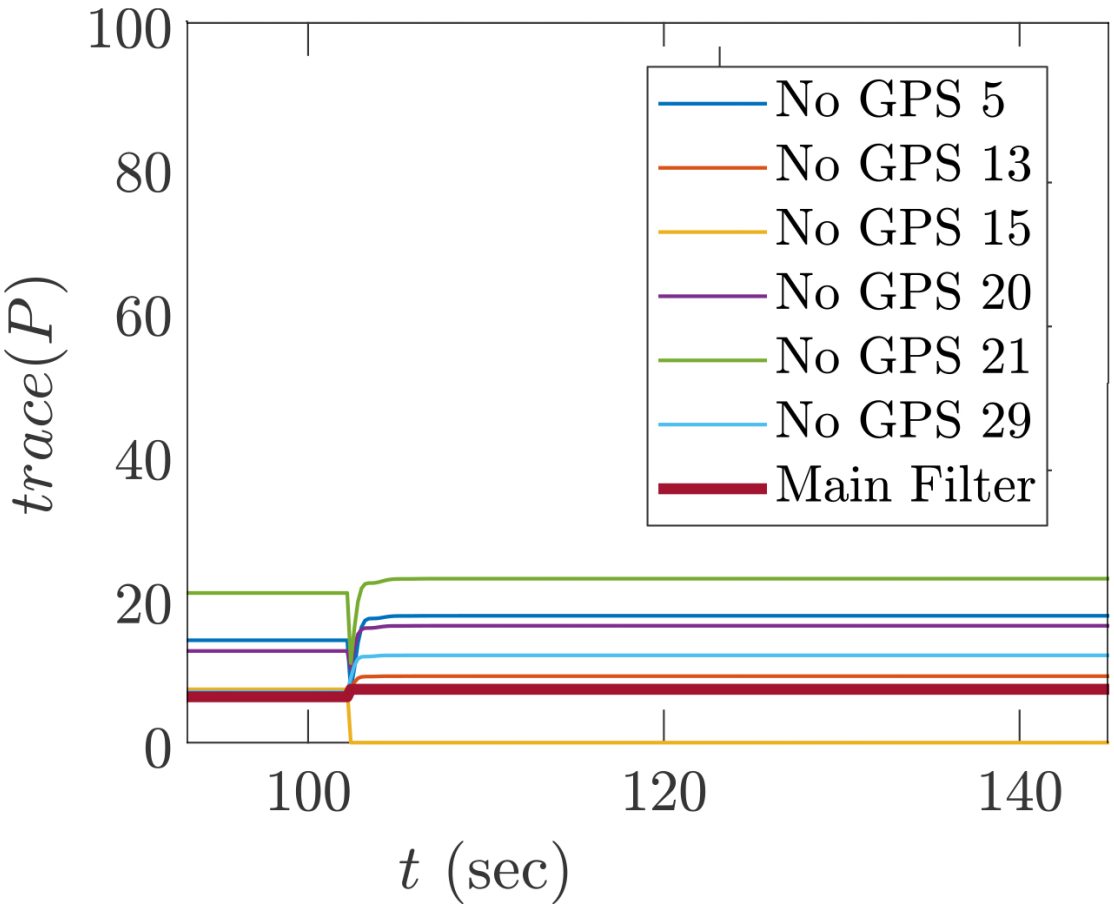

Figure 5.4: Trace of *a posteriori* Layer 1 Subfilter Position Covariance, Simulated Single GPS 15 Pseudorange Sensor Failure

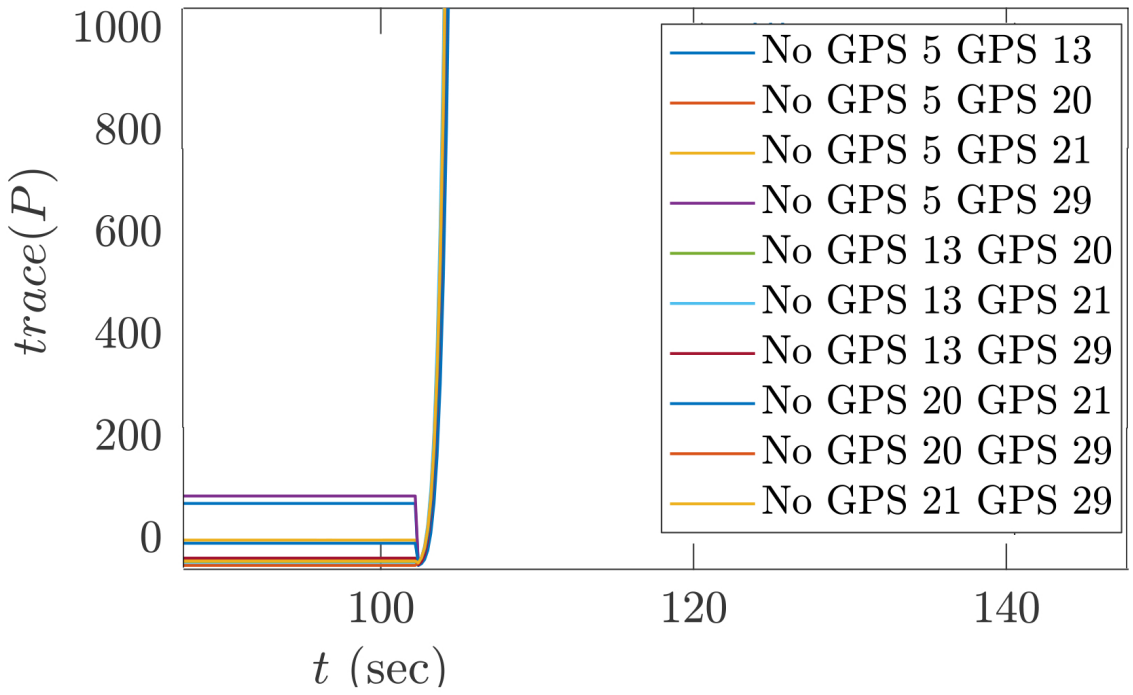

Figure 5.5: Trace of *a posteriori* Layer 2 Subfilter Position Covariance, Simulated Single GPS 15 Pseudorange Sensor Failure

For analysis, we configured ARMAS with a sensor package consisting of 6 individual pseudorange sensors, one for each visible SV. This means each filter in the FDE layer is equipped with a unique combination of 5 sensors. The latter half of the flight test dataset contains numerous pseudorange sensor dropouts. During analysis, it was observed that sensor dropouts tended to cause spurious behavior in ARMAS. This behavior occurs when less than 5 SVs are visible, meaning each layer 1 subfilter is unable to perform a stable position state estimate with less than 4 SVs. Further analysis reveals that SAARM is unable to provide an agreement to identify a failed sensor when the FDE subfilter layer loses overlapping position state observability. In other words, SAARM can detect a sensor fault but can not exclude the faulty sensor if even a single FDE layer subfilter loses position state observability due to dropout, poor geometry, etc. Since the initial simulation of ARMAS was performed with fully overlapping position state observability in the FDE layer, this deficiency was overlooked.

Consider a scenario where SV 'GPS 15' in a 6-SV constellation is excluded by ARMAS due to a simulated pseudorange bias at $t = 100$ sec. Figure 5.4 shows a local observability analysis for this scenario with the remaining observability subfilters, FDE layer subfilters, and Main Filter. Note there are 10 layer 2 subfilters remaining (from the original 15) in the observability layer, corrresponding to 5 remaining SVs (after GPS 15 was excluded). Note that each layer 2 subfilter in the observability layer is informed by exactly 4 unique pseudorange sensors until GPS 15 is removed. Since 4 unique pseudoranges are required to constrain a stable 3-D position solution with clock bias estimation, the unbounded position state covariance matrix indicates a complete loss of observability after $t = 100$ sec. As mentioned, a single pseudorange sensor dropout in this scenario will result in the loss of position state observability and is evidenced by an increase in the position state covariance estimate. If an additional sensor failure occurs, the decision-making FDE layer of ARMAS will struggle to provide a subfilter agreement to determine the culprit sensor

due to a reduction in position state observability. This analysis forms the genesis of local observability monitoring at the layer 2 "sub-subfilter" level to preserve the consistency of the layer 1 subfilter FDE and integrity functions of ARMAS.

5.3 Novel Observability Layer 2 "Sub-subfilter" Bank

In the previous section, we began the motivation for monitoring observability at a layer deeper than the FDE and integrity operations to preserve estimation consistency and framework resiliency. 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 [44]. 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* [44]. **If the states of interest in each layer 2 subfilter are observable and stabilizable, then each layer 1 subfilter inherits these properties.**

This means that the physical region encompassed by the position covariance estimates of all layer 1 subfilters contains the true navigation state within the statistical significance of the fault detection tests. That said, 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 5.6). Each unique subfilter in the observability layer excludes measurements from two sensors. 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.

For example, if a failed sensor is excluded, the FDE layer will be repopulated with new subfilters, each missing a single unique sensor and the failed sensor. Prior to the sensor exclusion, a subset of the observability layer contains the set of filters needed to spawn the new FDE subfilter layer after the exclusion. The purpose of monitoring the observability one layer deeper than the decision-making FDE layer is to provide a mechanism to warn the user in the event that a single sensor failure could jeopardize the consistency of the FDE and integrity operations. This warning comes in the form of a user "observability warning" to add an additional sensor to the framework. If only a single subfilter in the FDE layer loses overlapping position observability, SAARM is unable to provide a subfilter agreement required to identify and exclude the failed sensor. Once overlapping position observability is lost in the FDE layer, ARMAS can no longer guarantee at least one consistent uncorrupted subfilter which is required to preserve solution integrity. Due to potentially variable Fisher information available from all-source navigation sensors, it is critical that a resilient all-source navigation framework contains a method to monitor observability prior to corruption of the decision-making FDE layer.

To maintain resilience to F simultaneous sensor failures, the number of concurrent layer 2 subfilters in the observability filter bank, N, required to monitor observability for I sensors is

$$N = \binom{I}{F+1} = \frac{I!}{(F+1)(I-(F+1))!}. \tag{5.18}$$

As one might expect, the processing requirements to monitor each layer 2 subfilter position state covariance are non-trivial. For example, an 8 sensor system requires 28 concurrent SAARM layer 2 subfilters for resiliency to a single simultaneous sensor failure. When summed with the main filter and 8 traditional subfilters, a total of 37 concurrent estimation filters must be maintained. This method monitors overlapping position observability at the processing expense of factorial growth in the required quantity of concurrent estimation filters. A major benefit of this approach is evident in the event

of a sensor failure. Since a subset of the observability filter bank will form the new FDE layer, maintenance of these filters in the observability layer eliminates FDE and integrity operation downtime normally required during FDE layer re-initialization. Additionally, if we monitor the magnitude of the state estimate variance in each layer 2 subfilter we can determine which sensors provide critical Fisher information about our state(s) of interest prior to fault detection and exclusion events.

5.4 Stable Observability Monitoring (SOM)

The ability to maintain stable *a posteriori* estimates of system states is a primary indicator of overall estimator stability [31]. A primary goal of observability analysis is to measure the influence measurements have on system states [56]. Observability analysis has been applied to fused estimation with a variety of approaches including information matrix [35], error covariance analysis [73], and others [51] [33] [48]. It is well understood that a discrete linear time varying system is globally observable if the rank of its observability matrix $\mathbf{M}$ is full rank (i.e. same rank as quantity of states) for all time indices k. The degree of local observability can be defined as the measure of the singularity of $\mathbf{M}$ over a finite set of k [19]. For linear systems, the observability Gramian can be obtained as a solution of the the Lyapunov equation [30].

Directly related to the observability Gramian is the Fisher Information matrix which is a measure of the certainty of the state estimate due to measurement data alone [50] [59] [67]. The discrete recursive definition of the Fisher information matrix $\boldsymbol{F}$ is

$$\boldsymbol{F}(t_k^+) = \boldsymbol{\Phi}^T(t_k)\boldsymbol{F}(t_k^-)\boldsymbol{\Phi}(t_k) + \mathbf{H}^T(t_k)\mathbf{R}(t_k)^{-1}\mathbf{H}(t_k) \tag{5.19}$$

where the Fisher information contained in a single update at t_k is $\mathbf{H}^T(t_k)\mathbf{R}(t_k)^{-1}\mathbf{H}(t_k)$ which is the same term used to generate

$$\mathbf{P}(t_k^+)^{-1} = \mathbf{P}(t_k^-)^{-1} + \mathbf{H}^T(t_k)\mathbf{R}(t_k)^{-1}\mathbf{H}(t_k). \tag{5.20}$$

This relationship can be leveraged for observability analysis in a nonlinear estimator. If the system model is stochastically controllable and observable, then $\mathbf{P}(t_k^+)$ is uniformly bounded from above [56]. Stabilizable states have a unique positive-definite $\mathbf{P}(t_k^+)$ [56]. By monitoring post-update covariance matrices over time, we can ascertain if the signal to noise ratio in the system enables stabilized estimation. For the purposes of this chapter, we focus on the stability of the position states because we are particularly interested in preserving the consistency of the navigation integrity solution provided by the FDE sublayer in ARMAS.

The user-defined monitoring epoch for the sum of Mahalanobis distances in (5.13) adjusts the sensitivity of the SAARM monitoring test within the ARMAS framework [62]. This is a moving average of recent residual-space test statistics formed by pre-update residual vectors from (5.10) and (5.11). The length of 'monitoringTime' ($M\Delta t$) is an ARMAS tuning parameter which is used to adjust for detection sensitivity for temporal anomalies. The ARMAS 'monitoringTime' is designed to contain a sufficient quantity of samples to meet central limit theorem [65] criteria for the desired α, known as the Type I error rate (probability of false alarm). The pluggable estimation architecture provided by SCORPION enables propagation of multiple layers of subfilters. We record 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

$$O_{k,i}(n) = \begin{cases} 1, & \text{if } \mathbf{P}_j^{[n]}(t_k^+) > \mathbf{P}_j^{[n]}(t_{k-M}^+)\beta \\ 1, & \mathbf{P}_j^{[n]}(t_k^+) > \mathbf{P}_{j,max} \\ 0, & \text{otherwise} \end{cases} \tag{5.21}$$

where $\mathbf{P}_i^{[n]}(t_k^+)$ is the jth diagonal of the post-update state covariance matrix for layer 2 subfilter n, $\mathbf{P}_i(t_{k-M}^+)$ is the post-update variance for state estimate element j for layer 2 subfilter n exactly M samples prior to t_k, $\mathbf{P}_{j,max}$ a user-defined limit for maximum steady-state state estimate covariance, and $\beta \in [1, \infty)$ is the state estimate covariance transient growth threshold. If $k < M$ (e.g. a newly initialized filter), an observability flag is not set. A more compact form used to monitor position states is

$$O_{k,pos}(n) = \begin{cases} 1, & \text{if } tr(\mathbf{P}_{pos}^{[n]}(t_k^+)) > tr(\mathbf{P}_{pos}^{[n]}(t_{k-M}^+))\beta \\ 1, & \text{if } tr(\mathbf{P}_{pos}^{[n]}(t_k^+)) > tr(\mathbf{P}_{pos,max}) \\ 0, & \text{otherwise} \end{cases} \tag{5.22}$$

where $\mathbf{P}_{\mathbf{pos}}^{[\mathbf{n}]}(t_k^+)$ is the most recent post-update position covariance matrix for layer 2 subfilter n, $\mathbf{P}_{\mathbf{pos}}^{[\mathbf{n}]}(t_{k-M}^+)$ is the post-update position covariance matrix for layer 2 subfilter n exactly M samples prior to t_k $\mathbf{P}_{pos,max}$ is a user-defined limit for maximum steady-state position state estimate covariance, and $\beta \in [1, \infty)$ is the state estimate covariance transient growth threshold. 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 $tr(\mathbf{P}_{pos}^{[n]}(t_k^+))$ converges, then the individual position estimate variances also converge. When applying (5.22), it is important to ensure units are identical across the grouped states.

By measuring the difference between the trace of post-update position covariance matrices, we can determine if the position state information contained in the $\mathbf{H}^T(t_k)\mathbf{R}(t_k)^{-1}\mathbf{H}(t_k)$ terms in the previous M measurements have resulted in a stable mean estimate of all position elements in that subfilter. The user sets β as a tuning parameter for acceptable transient variance growth per monitoring epoch. 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 (5.23). The newly added sensor will directly en-

ter monitoring mode if it is considered 'trusted' or must pass through sensor validation if 'untrusted'.

$$SOM_{AddNewSensor} = \begin{cases} true, & \text{if } \sum_{n=1}^{N} O_k(n) > 0 \\ false, & \text{otherwise} \end{cases} \tag{5.23}$$

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. The results of (5.21) or (5.22) are ignored until the newly requested sensor completes an ARMAS monitoring period after entering monitoring mode. If post update variance stability is regained after the requisite ARMAS monitoring period, then (5.21) or (5.22) will set to 0 for each stable layer 2 subfilter and the observability warning is rescinded. This method flags the presence of an information deficiency with respect to the estimated states of interest in real-time across multiple layer 2 subfilters in a novel observability bank which is intended to preserve the consistency of the FDE and integrity operations in ARMAS.

5.5 State Estimate Covariance Transient Growth Threshold (Beta) and Maximum State Estimate Covariance Limits

The state estimate covariance transient growth threshold, $\beta \in [1, \infty)$, sets SOM framework sensitivity to transient loss of Fisher information for the state(s) of interest (5.21). The maximum state estimate covariance limit is simply an upper bound for state estimate covariance and is designed to set a minimum steady-state information threshold for the framework. The validation gate employed by SAARM is a moving window of residual-space test statistics in the form of Mahalanobis distances. The power of SAARM's chi-squared distributed hypothesis test is dependent on the ARMAS framework's ability to produce a stable and consistent pre-update residual covariance matrix, $\mathbf{\Sigma}$ (5.14). By

operating SOM's state estimate covariance monitoring scheme in the layer 2 sublayer (See Figure 5.6), we are able to peek forward at framework stability in the event of a single unknown sensor failure.

Since the pre-update residual covariance matrix is a function of the state estimate covariance and the observation model in (5.9), the post-update state estimate covariance $\mathbf{P}^+$ is a direct indicator for estimator observability and stabilizability [56]. The least stabilizable layer 2 subfilter is informed by exactly 1 fewer sensor than the least stabilizable layer 1 subfilter. By monitoring the layer 2 subfilter level, this allows for a sensor augmentation request before a potential loss of stability in the FDE operations. Since $\mathbf{T}_i$-matrix exclusion operations employed by SAARM require a unanimous fault agreement to exclude a failed sensor (5.17), it is particularly important that the stability of the layer 1 subfilter estimates is ensured. Furthermore, since ARMAS navigation integrity is provided by the union of the layer 1 subfilter position covariance ellipses, position integrity can be stabilized if the position states are selected as the SOM states of interest.

A selection of $\beta = 1.0$ is the most sensitive case which results in a request for an additional sensor when any layer 2 subfilter state estimate covariance grows by any amount in a single ARMAS monitoring epoch according to (5.23). This extreme case will maximize use of available offboard information. In the opposite case, large values of β will produce a less sensitive framework without a tendency to request additional sensor augmentation, where $\beta \to \infty$ performs identically to legacy ARMAS. For initial implementation, we recommend beginning with a small value for $\beta \approx 3.0$. This tuning parameter should be adjusted by the user to set the desired balance between autonomy and framework estimation stability for the state(s) of interest. The next section describes a set of reconfigurable GNSS scenarios used to assess the effectiveness of SOM.

5.6 Simulation

This section describes a set of four example 3-D scenarios designed to assess the impacts of SOM on ARMAS FDE operations and untrusted sensor validation. Individual SVs are arranged in random stationary GNSS constellations in a local tangential frame (LTF). In scenarios 1-4, we initialize three aircraft operating a standard EKF, ARMAS, and ARMAS-SOM (5.22) with 4, 5, 6, and 7 trusted pseudorange sensors, respectively. We assume the remaining untrusted SVs are available for augmentation via SOM. The availability of additional untrusted sensor information can be analagous to offboard augmentation in a collaborative navigation scenario. Each constellation contains 1 SV directly overhead and 9 SVs evenly distributed in azimuth with discrete random uniform elevations between 45 and 63.4 degrees (See Fig. 5.7). Each scenario consists of two sensor anomalies introduced at fixed times: (1) a growing pseudorange bias (linear ranging ramp) from $t = 240$ to $t = 330$ sec on a trusted sensor and (2) validation of an untrusted pseudorange sensor with a constant 40 meter bias at $t = 360$ sec. The growing pseudorange bias on the trusted sensor is used to assess FDE in monitoring mode (i.e. how large is the bias before it is detected and excluded). The constant bias is used to measure probability of detection in validation mode after recovery from a sensor exclusion.

Consider a 3-D example with two vehicles obtaining their navigation solutions from an EKF within the ARMAS framework

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{\mathbf{x}}_p(t) \\ \dot{\mathbf{x}}_v(t) \\ \dot{\mathbf{x}}_a(t) \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{x}}_v(t) \\ \dot{\mathbf{x}}_a(t) \\ -\frac{1}{\tau_a}\dot{\mathbf{x}}_v(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \mathbf{w(t)} \end{bmatrix} \tag{5.24}$$

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 $\tau_a = 90$ seconds is a time-constant associated with a First-order Gauss-Markov (FOGM) process. A 3-D white noise process is given by $\mathbf{w}(t)$

where $E[\mathbf{w}(t)\ \mathbf{w}(t+\tau)^T] = \mathbf{Q}\delta(\tau)$ and

$$\mathbf{Q} = (1.0 \times 10^{-2})^2 \mathbf{I}_{3\times 3}\ (m^2/s^4) \tag{5.25}$$

The model from (5.24) is used to generate randomly initialized vehicle trajectories for each trial. Initial velocities and accelerations are Normally distributed with $\sigma_{accel} = 1\times10^{-2}$ (m/s^2) and $\sigma_{vel} = 5$ (m/s). Figure 5.8 shows 300 sample truth trajectories for this scenario.

Each vehicle is initialized identically with initial state and covariance estimates:

$$\hat{\mathbf{x}}(0) = \begin{bmatrix} 0 & 0 & 200 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T, \tag{5.26}$$

$$\mathbf{P}(0) = diag\left(\begin{bmatrix} 30^2 & 30^2 & 30^2 & 10^2 & 10^2 & 10^2 & 0.01^2 & 0.01^2 & 0.01^2 \end{bmatrix} \right). \tag{5.27}$$

Each aircraft receives discrete measurements from a constellation of stationary satellite vehicles (SV). Although there are 10 SVs (labelled GPS1-10), the aircraft are initialized with a trusted subset of this constellation varying from 4 to 7 SVs. An additional fixed satellite with a random uniform elevation between 45 and 63.4 degrees is introduced at $t = 360$ sec and is corrupted with a constant 40 meter pseudorange range bias. This constellation is designed to provide coverage at high elevation angles between approximately 45 degrees and 63.4 degrees with a single satellite directly overhead (Figure 5.7).

Individual pseudorange measurements are performed according to (5.28)

$$\rho_i = \sqrt{(X_{SV,i} - X_u) + (Y_{SV,i} - Y_u) + (Z_{SV,i} - Z_u)} + b_u, \tag{5.28}$$

where ρ_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 $\mathbf{R}_{SV} = 10^2$ m^2. A receiver clock bias b_u is

independently estimated as an additional state in each EKF. All coordinates are expressed in the LTF.

Pseudorange measurements, $\hat{\mathbf{z}} = h(\hat{\mathbf{x}})$, are performed according to (5.30) for the estimated position states (X_u, Y_u, Z_u, b_u) within $\hat{\mathbf{x}}$. The measurement Jacobian $\boldsymbol{H}$ is:

$$\boldsymbol{H} = \begin{bmatrix} \frac{-(X_{i,SV}-X_u)}{r_i} & \frac{-(Y_{i,SV}-Y_u)}{r_i} & \frac{-(Z_{i,SV}-Z_u)}{r_i} & 1 \\ \dots & \dots & \dots & 1 \\ \frac{-(X_{n,SV}-X_u)}{r_n} & \frac{-(Y_{n,SV}-Y_u)}{r_n} & \frac{-(Z_{n,SV}-Z_u)}{r_n} & 1 \end{bmatrix} \tag{5.29}$$

where

$$r_i = \sqrt{(X_{SV,i} - X_u) + (Y_{SV,i} - Y_u) + (Z_{SV,i} - Z_u)} \tag{5.30}$$

is the distance from the platform to an SV.

In addition to initial pseudorange sensors, additional unbiased pseudorange sensors may be requested automatically by SOM at any time during the scenario. The other two approaches (legacy ARMAS, EKF) are limited to the sensors provided during initialization. In this respect, SOM has a clear advantage over the two legacy approaches. The point of this analysis is to show that ARMAS-SOM can detect a threat to navigation resilience, provide timely sensor augmentation, and successfully preserve the consistency of the FDE and validation operations in ARMAS. In the middle of the scenario, an insidious growing position bias is injected into a single trusted pseudorange sensor to test the ARMAS FDE process. The size of the position bias at exclusion is recorded. Near the end of the scenario, an untrusted pseudorange sensor with a fixed bias is added into validation at $t = 360$ sec to test ARMAS sensor validation.

5.7 Numerical Results

The following section presents the results for three identical aircraft (EKF, ARMAS, and ARMAS-SOM equipped) in four scenarios with 4, 5, 6, and 7 initial trusted

pseudorange sensors. We assume the remaining untrusted SVs are available for augmentation via SOM. The maximum position state estimate covariance threshold is set to $\mathbf{P}_{pos,max} = diag_{3\times3}(20^2 \text{ m}^2)$ (5.22). The acceptable steady state variance growth term β from (5.22) is set to 3.0 for all runs. This was achieved by initially setting β to 1.0 and incrementally increasing until desired sensitivity is achieved.

In each scenario, a single trusted pseudorange sensor experiences a sensor anomaly (linear ramp) from $t = 240$ to 330 sec and an untrusted biased sensor (40 meters) is added at $t = 360$ sec. The EKF simply trusts all information provided, has no ability to detect faults nor perform sensor validation, and is included only as a performance baseline. ARMAS is equipped with FDE capabilities and validation of untrusted sensors. The ARMAS recalibration and remodeling modes are not active. ARMAS-SOM includes all of the aforementioned ARMAS capabilities and adds the ability to request additional untrusted pseudorange sensors at any time. SOM monitors the stability of the state covariance estimates in the observability subfilters (layer 2) (5.22). Since each observability layer subfilter exclude 2 sensors, the minimum number of pseudorange sensors at the user output level (layer 0) is 6. If one of the trusted sensors is excluded, the quantity of sensors at the top level must remain at least 6 to ensure stability at the layer 2 subfilter level.

5.7.1 Scenario 1.

In scenario 1, each aircraft is initialized with 4 trusted pseudorange sensors receiving information from a random stationary constellation of SVs in the LTF. A summary of the results is shown in Table 5.1. Clearly, ARMAS-SOM outperforms both ARMAS and the EKF, especially in terms of RSS error and detection rate for a biased sensor. This is expected because ARMAS-SOM augments itself with 2 additional untrusted pseudorange sensors using the ARMAS validation process prior to the sensor anomaly event at $t = 240$ sec. Once the spoofed sensor is excluded, ARMAS-SOM requests one additional untrusted sensor to stabilize the observability layer prior to validation of the untrusted biased sensor.

This results in a total of 3 added pseudorange sensors for a total of 7. With ARMAS, the exclusion of the spoofed sensor results in only 3 pseudorange sensors which is insufficient to provide a stable 3-D position estimate with clock bias estimation. This is evidenced by the large growth in estimated standard error at approx. $t = 245$ sec in Figure 5.9. Figure 5.10 shows the mean RSS error and standard deviation for 1000 Monte Carlo trials. It is clear that ARMAS does not recover well from the exclusion of the spoofed sensor with a bias detection rate of 0.01 is clearly outperformed by ARMAS-SOM with a detection rate of 0.99. The EKF simply trusts all information provided and the navigation solution is carried off by the linear ramp sensor anomaly.

4 Trusted SVs, 1000 Trials					
Algorithm	**Grand Mean RSS Error [m] (Median)**	**Std. Dev. [m]**	**Mean GPS4 Anomaly Magnitude at Exclusion [m]**	**Detection Rate for 40 meter Bias**	**Mean Total Added Sensors**
EKF	916.0 (76.53)	1,720.8	–	–	–
ARMAS	694.3 (76.52)	684.3	44.53	0.01	–
SOM	16.78 (10.15)	18.25	41.51	0.99	3.00

Table 5.1: Scenario 1 Results: RSS Error, GPS4 Exclusion Magnitude, Detection Rate, and Quantity of Augmented Sensors

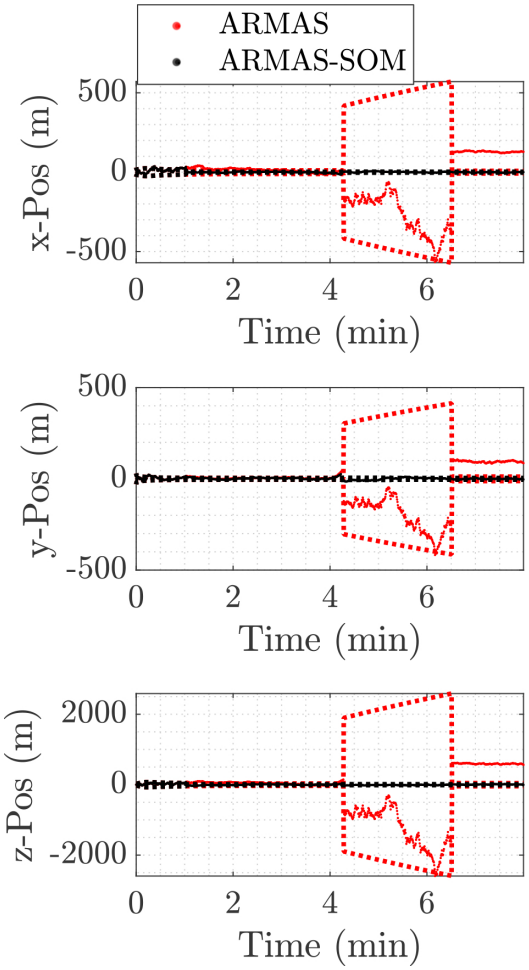

Figure 5.9: Scenario 1 State Estimation Error for 1 Run with Sensor Anomaly from t = 240 to 330 sec and Biased Sensor added at t = 360 sec

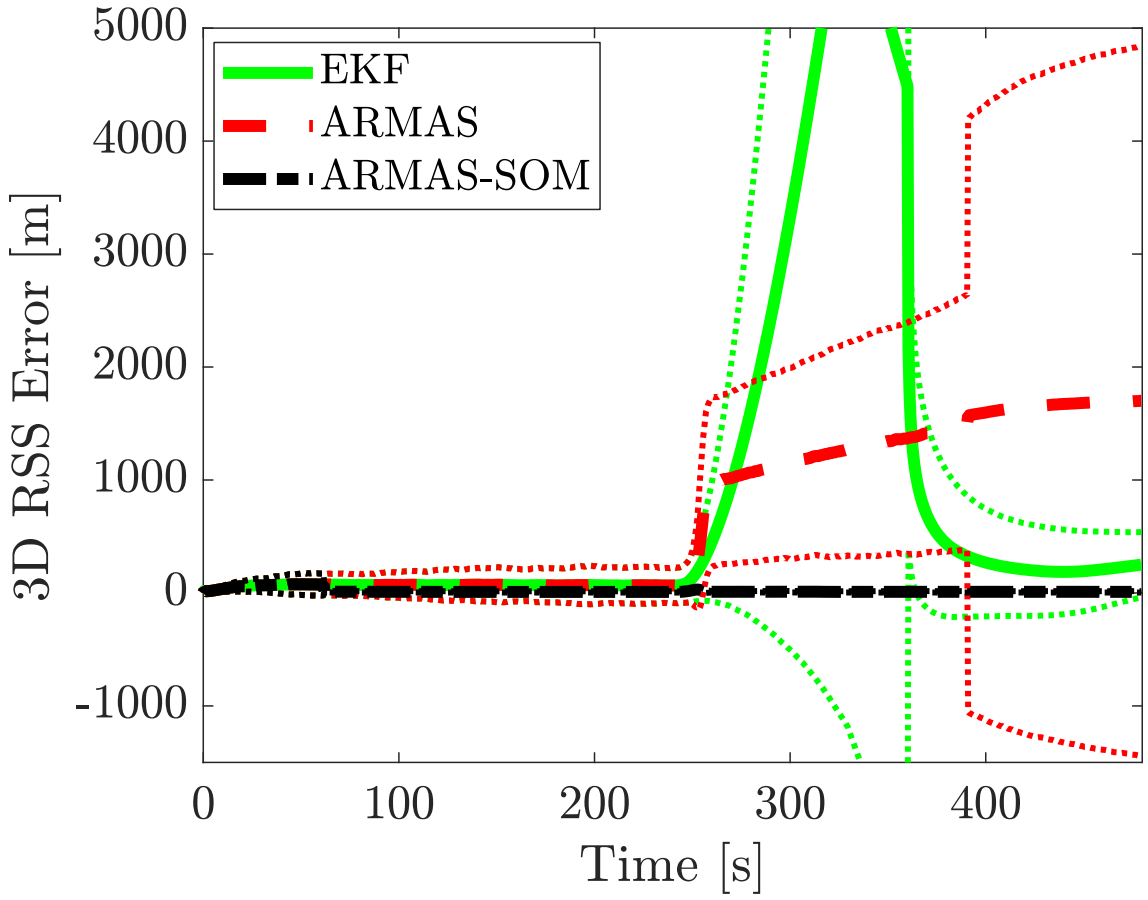

Figure 5.10: Scenario 1 Mean 3-D RSS Error ± 1-σ for 1000 Runs with Sensor Anomaly from t = 240 to 330 sec and Biased Sensor added at t = 360 sec

5.7.2 Scenario 2.

In scenario 2, each aircraft is initialized with 5 trusted pseudorange sensors. A summary of the results is shown in Table 5.2. ARMAS-SOM outperforms both ARMAS and the EKF. Figure 5.12 shows the improvement in estimation performance achieved by ARMAS-SOM sensor augmentation at approximately t = 70 sec. This is also visible at approximately t = 70 sec with the estimated standard error estimates in Figure 5.11. Once the spoofed sensor is excluded, ARMAS-SOM requests one additional untrusted sensor to stabilize the observability layer prior to validation of the unstrusted biased sensor. This

results in a total of 2 added pseudorange sensors for a total of 7. With ARMAS, the exclusion of the spoofed sensor results in 4 pseudorange sensors which is sufficient to provide a stable 3-D position estimate with clock bias estimation at the main filter level. Figure 5.10 shows the mean RSS error and standard deviation for 1000 Monte Carlo trials. It is clear that ARMAS and ARMAS-SOM performance is closer and while ARMAS maintains a stable solution, it has difficulty with proper validation of the untrusted biased sensor with a detection rate of 0.58 and is outperformed by ARMAS-SOM with a detection rate of 0.99. The EKF simply trusts all information provided and the navigation solution is exploited by the biased information.

5 Trusted SVs, 1000 Trials					
Algorithm	**Grand Mean [m] (Median)**	**Std. Dev. [m]**	**Mean GPS4 Anomaly Magnitude at Exclusion [m]**	**Detection Rate for 40 meter Bias**	**Mean Total Added Sensors**
EKF	283.5 (36.42)	411.3	–	–	–
ARMAS	39.07 (22.44)	34.22	40.36	0.58	–
SOM	12.14 (9.48)	7.25	37.93	1.00	2.00

Table 5.2: Scenario 2 Results: RSS Error, GPS4 Exclusion Magnitude, Detection Rate, and Quantity of Augmented Sensors

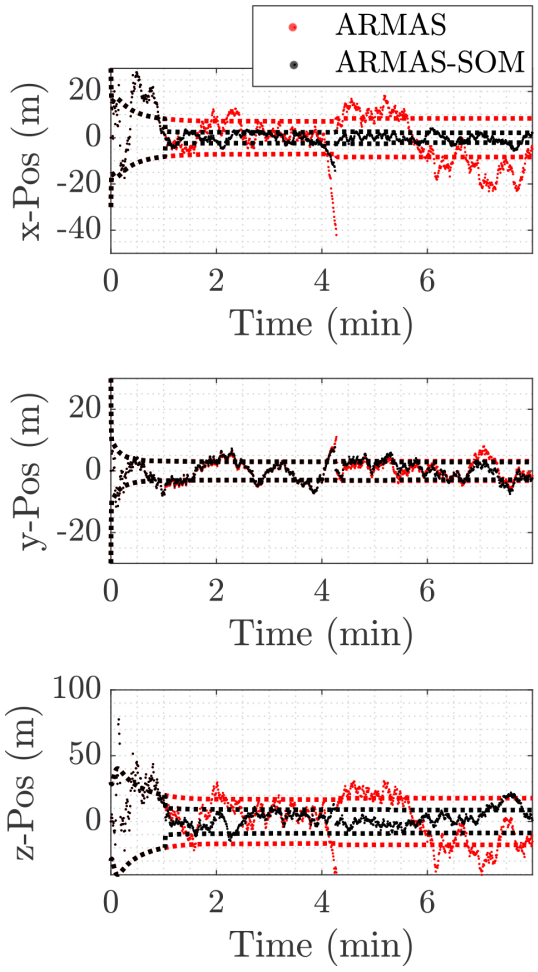

Figure 5.11: Scenario 2 State Estimation Error for 1 Run with Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec

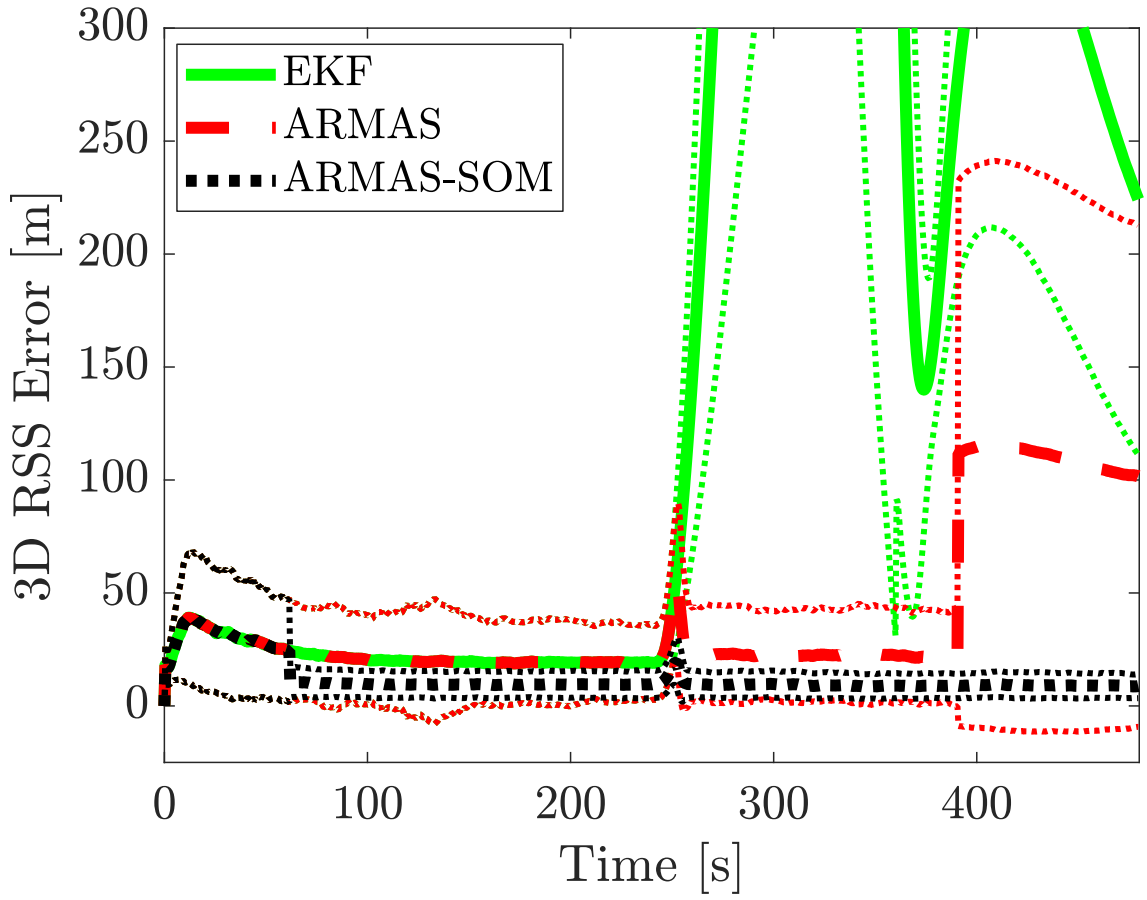

Figure 5.12: Scenario 2 Mean 3-D RSS Error ± 1-σ for 1000 Runs with Sensor Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec

5.7.3 Scenario 3.

In scenario 3, each aircraft is initialized with 6 trusted pseudorange sensors. A summary of the results is shown in Table 5.3. ARMAS-SOM outperforms both ARMAS and the EKF. Figures 5.13 and 5.14 show that ARMAS and ARMAS-SOM performance is nearly identical until ARMAS-SOM augments with an additional untrusted sensor at approximately t = 310 sec after spoofed sensor exclusion. This results in a total of 1 added pseudorange sensor for a total of 7. With ARMAS, the exclusion of the spoofed sensor results in 5 pseudorange sensors at the main filter level which is sufficient to provide

a stable 3-D position estimate with clock bias estimation in each layer 1 subfilter (each containing 4 pseudorange sensors). With an increase in information, it is clear that ARMAS performance is closer to ARMAS-SOM than in Scenario 2. ARMAS maintains a stable solution but has difficulty with proper validation of the untrusted biased sensor with a detection rate of 0.85 and is outperformed by ARMAS-SOM with a detection rate of 1.00. The EKF simply trusts all information provided and the navigation solution is exploited by the biased information.

6 Trusted SVs, 1000 Trials					
Algorithm	**Grand Mean [m] (Median)**	**Std. Dev. [m]**	**Mean GPS4 Anomaly Magnitude at Exclusion [m]**	**Detection Rate for 40 meter Bias**	**Mean Total Added Sensors**
EKF	153.7 (25.11)	213.6	–	–	–
ARMAS	17.73 (12.94)	9.00	38.44	0.85	–
SOM	12.34 (11.76)	3.97	38.44	1.00	1.00

Table 5.3: Scenario 3 Results: RSS Error, GPS4 Exclusion Magnitude, Detection Rate, and Quantity of Augmented Sensors

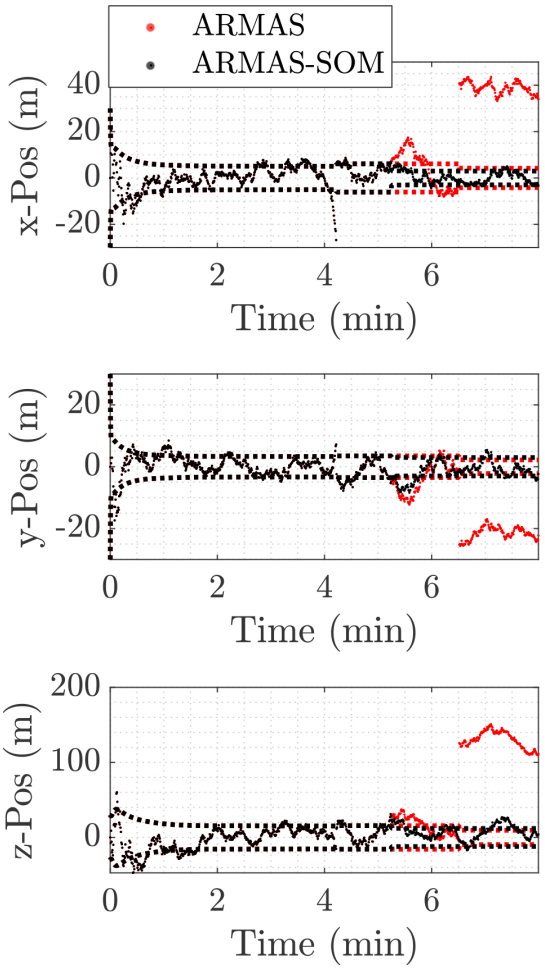

Figure 5.13: Scenario 3 State Estimation Error for 1 Run with Sensor Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec

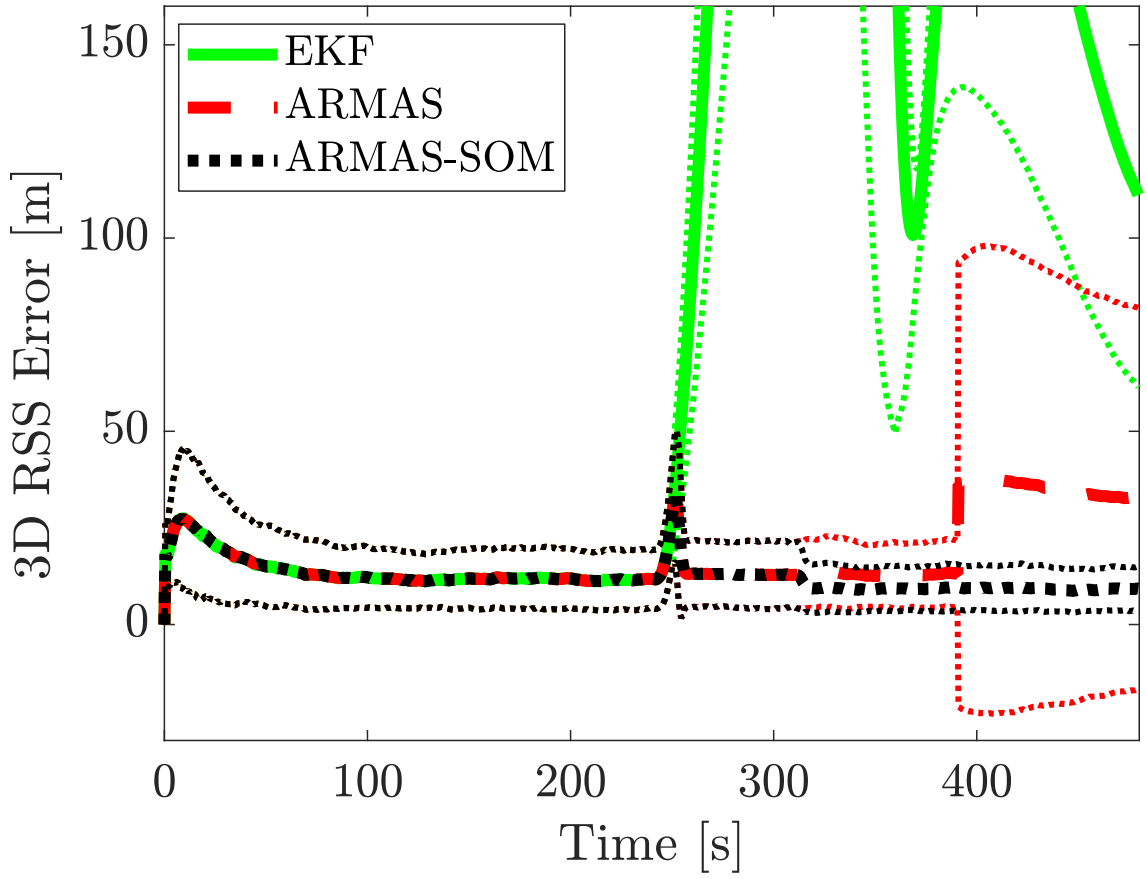

Figure 5.14: Scenario 3 Mean 3-D RSS Error ± 1-σ for 1000 Runs with Sensor Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec

5.7.4 Scenario 4.

In scenario 4, each aircraft is initialized with 7 trusted pseudorange sensors. A summary of the results is shown in Table 5.4. ARMAS-SOM performs identically to ARMAS and both outperform the EKF. Figures 5.15 and 5.16 show that ARMAS and ARMAS-SOM performance is practically identical. ARMAS-SOM does not request any sensor augmentation for a total of 7 pseudorange sensors. The EKF simply trusts all information provided and the navigation solution is exploited by the biased information.

7 Trusted SVs, 1000 Trials					
Algorithm	**Grand Mean [m] (Median)**	**Std. Dev. [m]**	**Mean GPS4 Anomaly Magnitude at Exclusion [m]**	**Detection Rate for 40 meter Bias**	**Mean Total Added Sensors**
EKF	85.52 (20.5)	113.4	–	–	–
ARMAS	11.04 (10.08)	2.55	36.75	0.99	–
SOM	11.04 (10.08)	2.55	36.75	0.99	0.00

Table 5.4: Scenario 4 Results: RSS Error, GPS4 Exclusion Magnitude, Detection Rate, and Quantity of Augmented Sensors

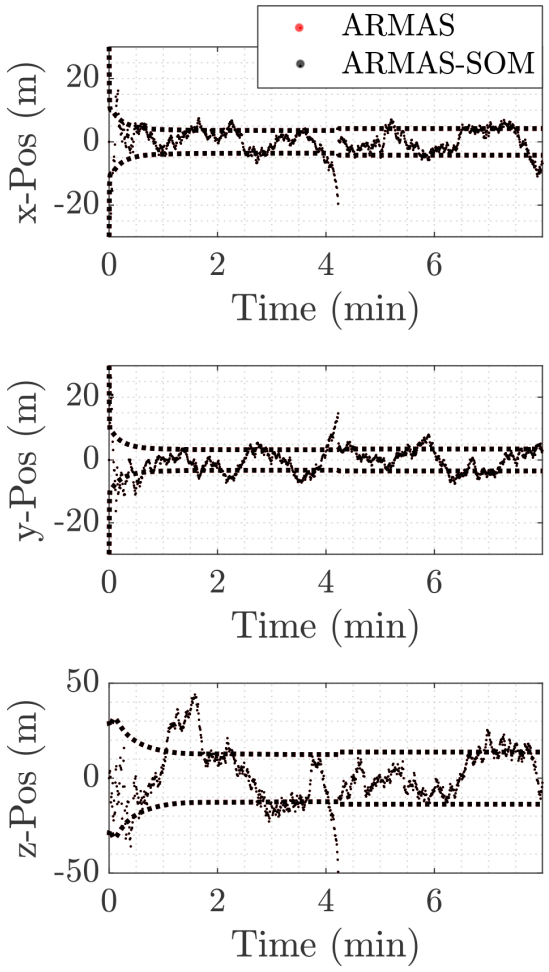

Figure 5.15: Scenario 4 State Estimation Error for 1 Run with Sensor Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec

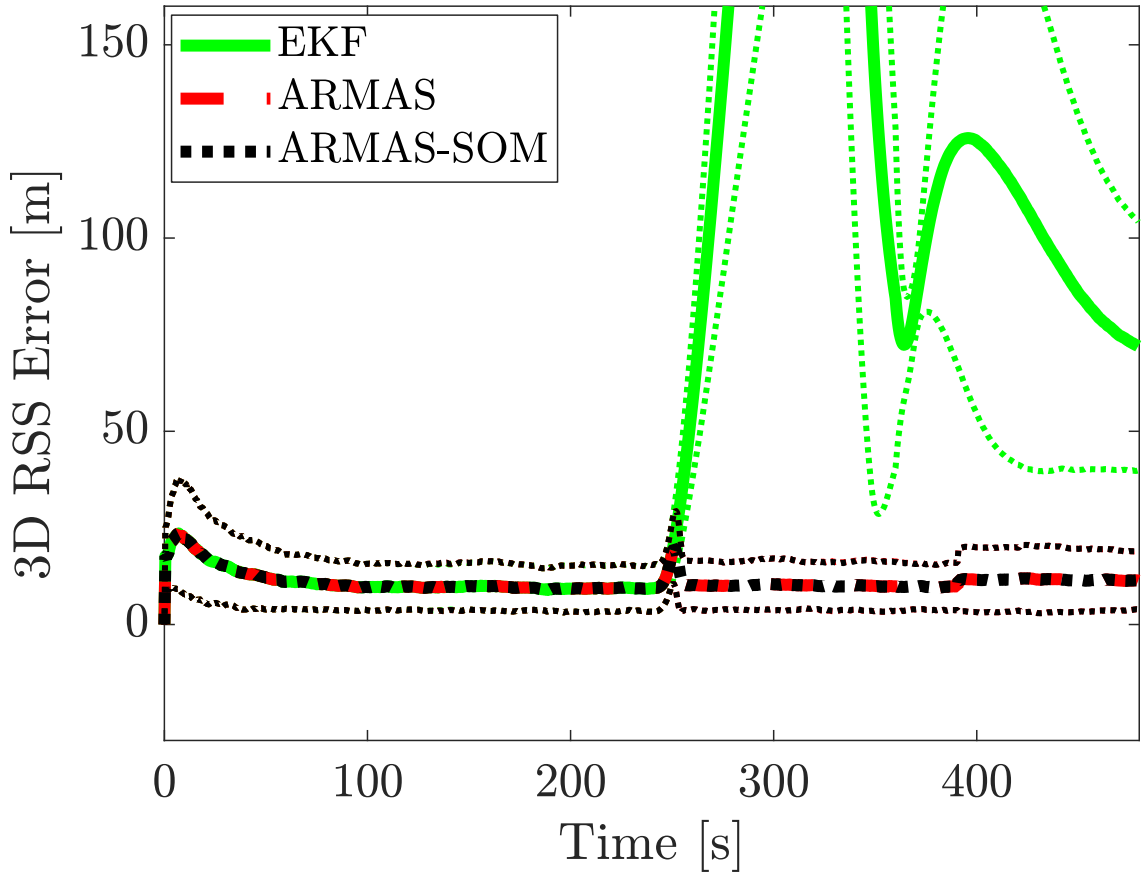

Figure 5.16: Scenario 4 Mean 3-D RSS Error ± 1-σ for 1000 Runs with Sensor Linear Pseudorange Ramp Bias from t = 240 to 330 sec and Biased Sensor added at t = 360 sec

5.7.5 Across Scenarios.

Figure 5.17 shows a summary of detection rates for each of the four 1000-run scenarios. Clearly, ARMAS validation consistency is a function of stable observability in the FDE (layer 1) subfilters. In order guarantee resilience to a single simultaneous sensor failure, stable observability must be established at the Observability (layer 2) subfilters.

5.8 Chapter Summary and Future Work

This chapter addresses a critical vulnerability of ARMAS and provides a convenient method to monitor real-time navigation resilience and eliminate subfilter respawn

downtime in the event of a sensor failure. This method presents a novel "observability bank" operating in a layer of I choose 2 subfilters. These additions to ARMAS provide real-time stable observability analysis via monitoring of the layer 2 subfilter a posteriori covariance matrices.

The ARMAS framework was originally developed with linear 2D position and velocity sensors which provided fully overlapping position observability. Initial analysis of ARMAS with GNSS pseudorange data from a sUAS flight test at Camp Atterbury, IN showed that ARMAS operations can become inconsistent if the FDE layer subfilters lose overlapping position estimation observability. SOM monitors each layer 2 subfilter for both observability and stabilizability.

To maintain resilience to a single simultaneous sensor failure, we must assume that a single sensor may fail at any time. Since the "observability" bank contains a subset of subfilters which will form the new FDE layer after a sensor exclusion, the observable and stabilizable properties guaranteed by SOM are inherited by the newly formed FDE layer. Furthermore, SOM provides the user with a timely warning to augment with additional sensor data and provides a notification when the augmented sensor information is sufficient for resilience to a single simultaneous sensor failure. A Monte Carlo analysis of four example scenarios proves that a loss of overlapping position observability in the FDE layer can result in an inability to exclude a failed sensor and inadvertent validation of a corrupted sensor, resulting in undetected corruption of the main navigation solution. SOM is shown provide instrinsic awareness about underlying overlapping observability assumptions made by ARMAS. With sensor augmentation, these assumptions can be preserved to guarantee ARMAS framework resilience to a single simultaneous sensor failure and is proven by the preservation of the ARMAS FDE and validation processes.

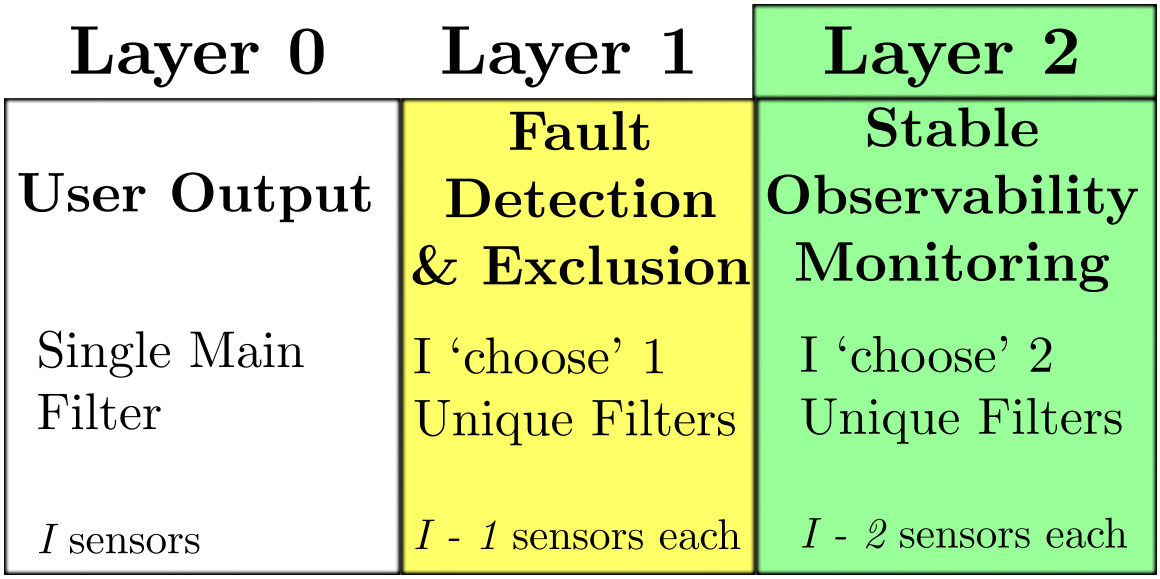

Figure 5.6: ARMAS Framework with Novel Observability Layer for Resiliency to One Simultaneous Sensor Failure.

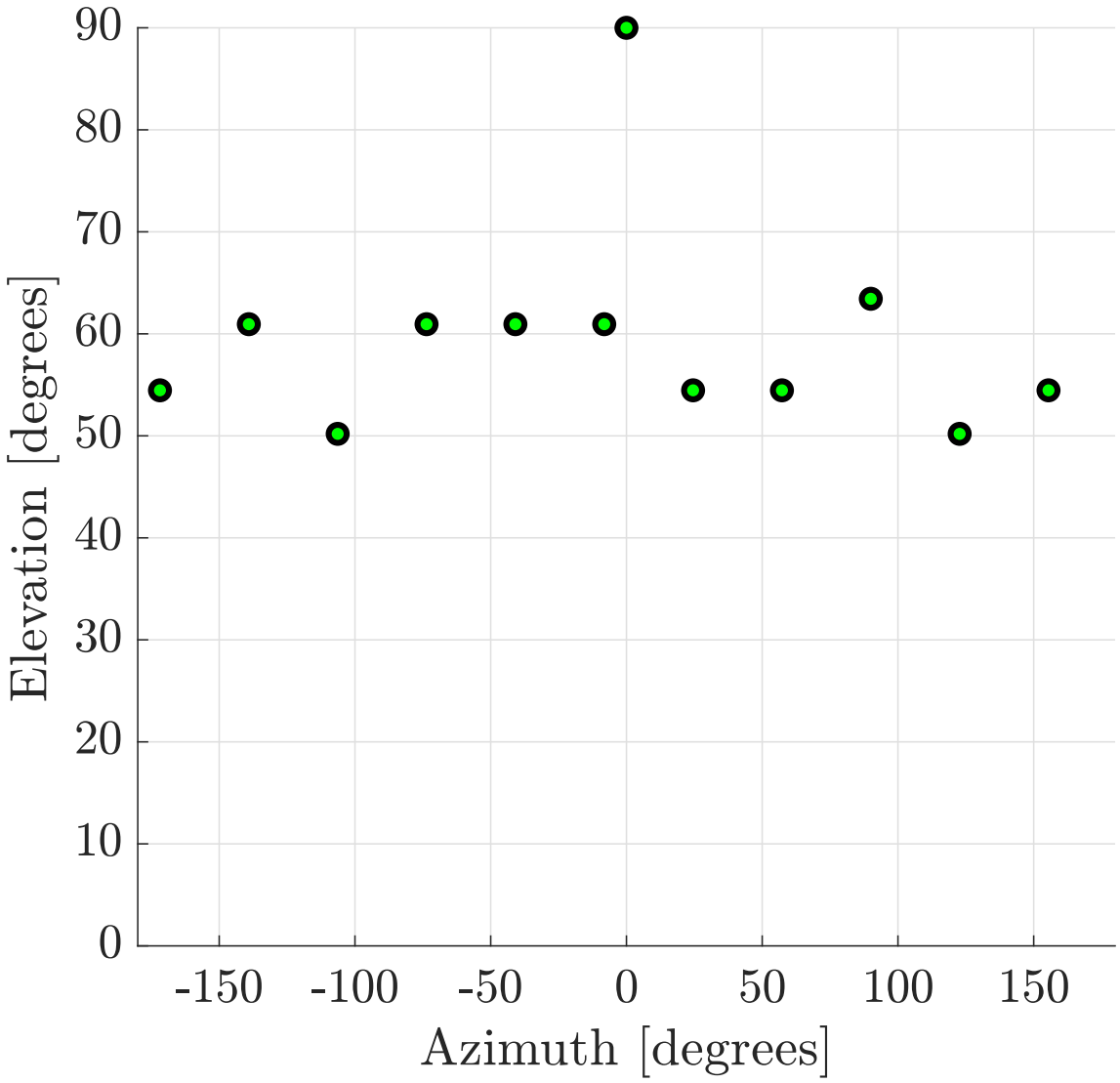

Figure 5.7: Sample of 10-SV Stationary Constellation Skyplot with Discrete Random Uniform Elevations and 1 Satellite Directly Overhead

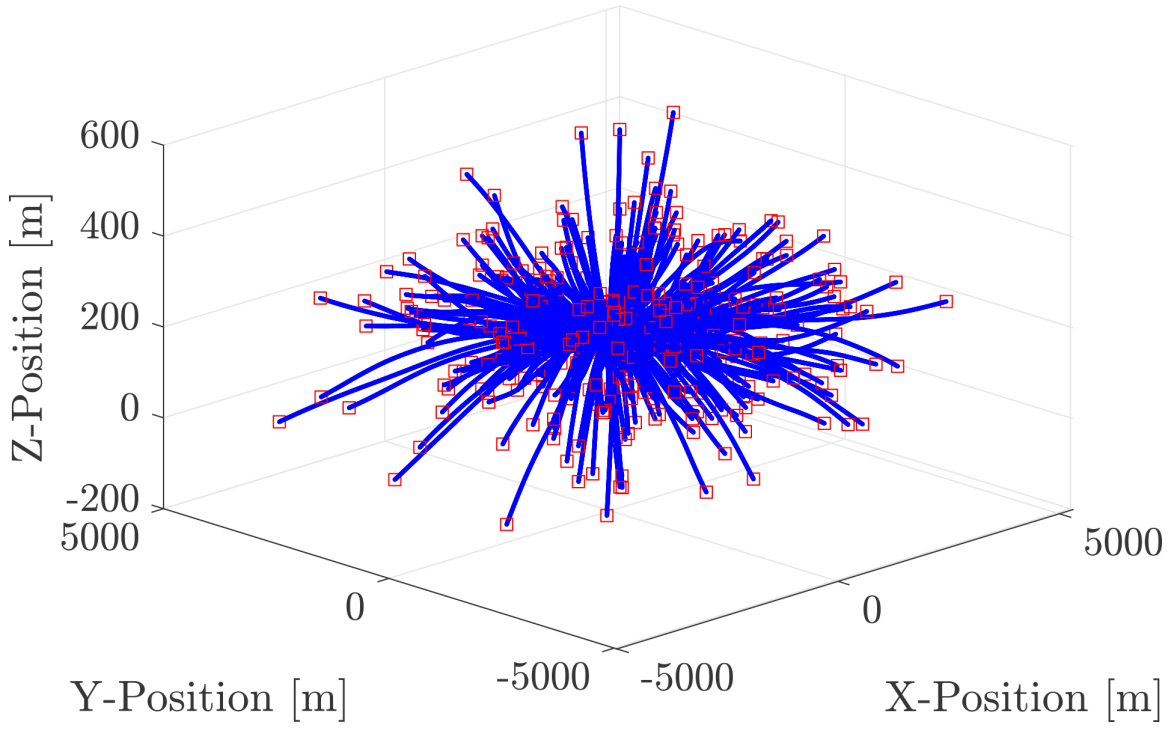

Figure 5.8: Sample Truth Trajectories, 300 Runs

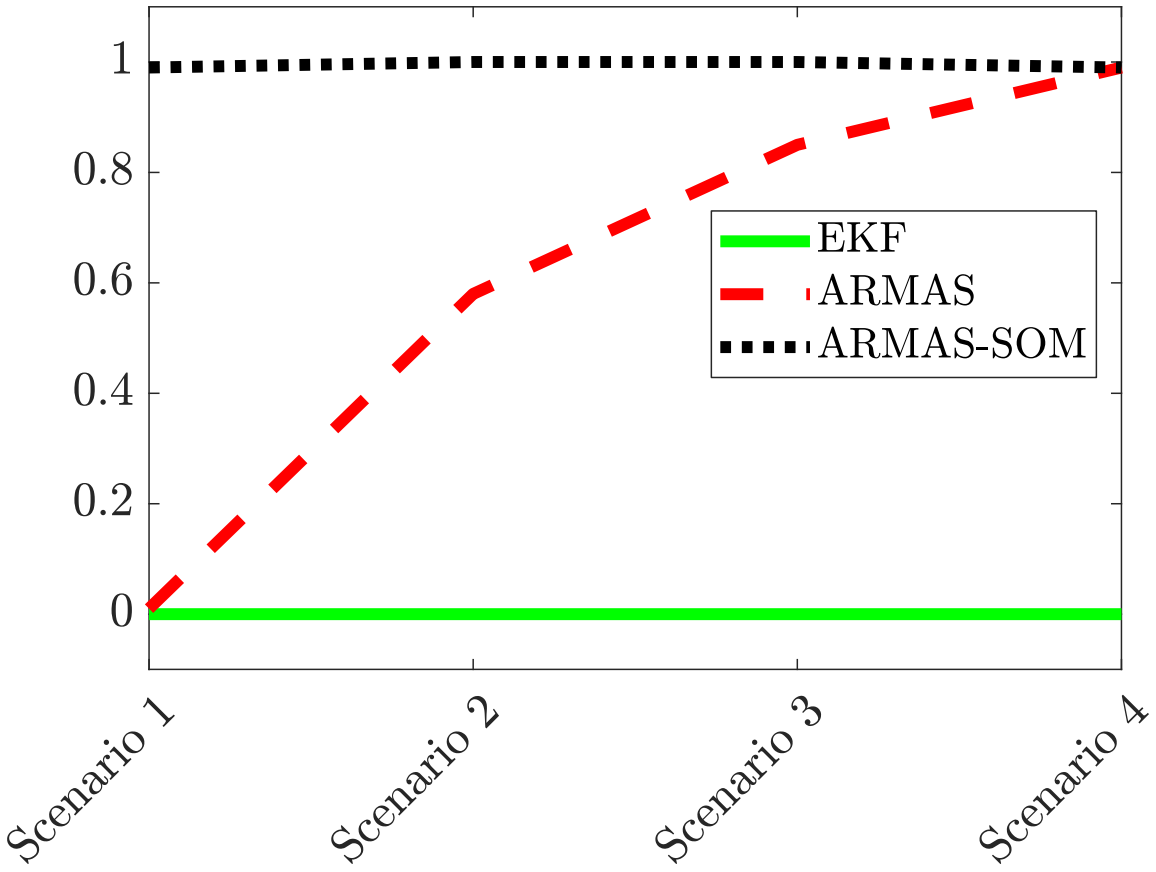

Figure 5.17: Summary of Detection Rates for 40 meter Biased Pseudorange Sensor

VI. A Framework for Resilient Collaborative All-source Navigation

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" 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 of ARMAS-SOM 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.

6.1 Introduction

Due to widespread awareness of GNSS vulnerabilities, there is increased interest in alternative navigation technologies such as visual [77], signals of opportunity [20], magnetic [13], and others [23][63][22]. The fusion of these information sources for localization is known as *all-source navigation* [74]. 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* [44] which is

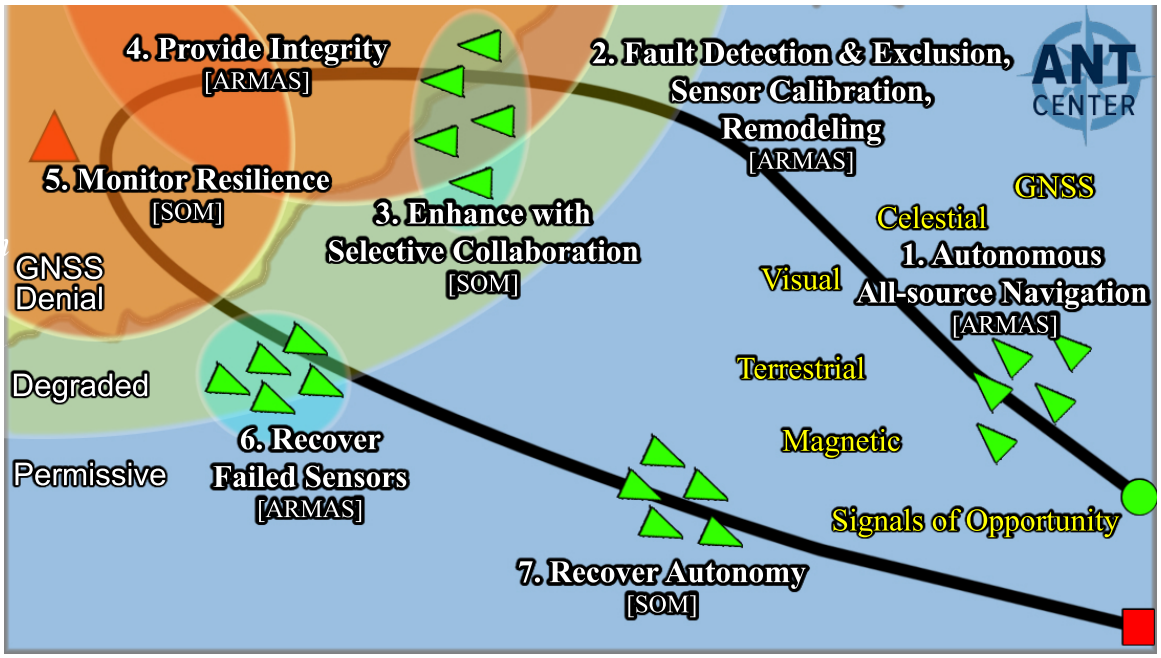

Figure 6.1: The ARMAS-SOM framework provides intrinsic all-source resilience "awareness" via novel selective offboard collaboration which improves navigation integrity and facilitates graceful recovery to autonomy.

currently provided through GNSS techniques such as Receiver Autonomous Integrity Monitoring (RAIM) [8]. With GNSS navigation, simultaneously redundant, synchronous measurements with validated measurement models are readily available. With all-source navigation, none of these are guaranteed. In 2018, ARMAS [43] was introduced to provide a generalized framework to fuse all-source sensor information which leveraged SCORPION pluggable Bayesian estimators to gracefully manage multiple concurrent state estimation subfilters [46]. In 2019, the monitoring mode of the ARMAS framework was redesigned to provide a means for all-source integrity, known as SAARM [44].

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 [43] which is augmented by Stable Observability Monitoring (SOM) [25]. 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.

6.2 Background

6.2.1 Recursive Bayesian Estimation.

Small cumulative measurement errors form an integration bias or “drift”, the primary error source for an 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 KF algorithm [45], presented in 1960. 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

$$\dot{\mathbf{x}}(t) = \mathbf{F}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t) + \mathbf{G}\mathbf{w}(t), \tag{6.1}$$

with

$$E[\mathbf{w}(t)] = 0, \tag{6.2}$$

$$E[\mathbf{w}(t)\mathbf{w}(t)^T(t+\tau] = \mathbf{Q}\delta(\tau), \tag{6.3}$$

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. E is the expectation operator, t is system time, and τ is a time offset. $\mathbf{F}$, $\mathbf{B}$, and $\mathbf{G}$ are linear operator matrices for the state vector, control input vector, and noise vector, respectively.

In 1978, the Van Loan method was introduced [76] to provide a method to discretize 6.1 and 6.3 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, Δt.

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

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \tag{6.4}$$

where

$$E[\mathbf{v}_k] = 0, \tag{6.5}$$

$$E[\mathbf{v}_k \mathbf{v}_{k+\Delta t}^T] = \mathbf{R}\delta_{k,k+\Delta t}, \tag{6.6}$$

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

$$\hat{\mathbf{x}}_{k+1}^- = \mathbf{\Phi}\hat{\mathbf{x}}_k^+ + \mathbf{B}_d\mathbf{u}_k, \tag{6.7}$$

$$\mathbf{P}_{k+1}^- = \mathbf{\Phi}\mathbf{P}_k^+\mathbf{\Phi}^T + \mathbf{Q}_d. \tag{6.8}$$

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

$$\mathbf{K}_k = \mathbf{P}_k^-\mathbf{H}^T[\mathbf{H}\mathbf{P}_k\mathbf{H}^T + \mathbf{R}]^{-1}, \tag{6.9}$$

$$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k[\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^-], \tag{6.10}$$

$$\mathbf{P}_k^+ = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\mathbf{P}_k^-. \tag{6.11}$$

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.

6.2.2 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 [56]. 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 SAARM algorithm implemented by the 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

$$\mathbf{r}_k = \mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^-. \tag{6.12}$$

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

$$\mathbf{r}_k \hookrightarrow \mathcal{N}\left(\mathbf{0}, \mathbf{H}\mathbf{P}_k^- \mathbf{H}^{\mathrm{T}} + \mathbf{R}\right), \tag{6.13}$$

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

$$p(\mathbf{r}_k|\mathbf{\Sigma}_k) = \frac{1}{\sqrt{|2\pi\mathbf{\Sigma}_k|}} e^{-\frac{1}{2}(\mathbf{r}_k)^{\mathrm{T}}\mathbf{\Sigma}_k^{-1}(\mathbf{r}_k)} \text{ and} \tag{6.14}$$

$$\mathbf{\Sigma}_k = \mathbf{H}\mathbf{P}_k^- \mathbf{H}^{\mathrm{T}} + \mathbf{R}. \tag{6.15}$$

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

$$L_{N_k} = \prod_{j=k-N+1}^{k} \frac{1}{\sqrt{|2\pi\mathbf{\Sigma}_j|}} e^{-\frac{1}{2}(\mathbf{r}_j)^{\mathrm{T}}\mathbf{\Sigma}_j^{-1}(\mathbf{r}_j)}. \tag{6.16}$$

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

$$\mathcal{L}_{N_k} = \sum_{i=k-N+1}^{k} -\log(|2\pi\mathbf{\Sigma}_i|) - \frac{1}{2}\mathbf{r}_i^{\mathrm{T}}\mathbf{\Sigma}_i^{-1}\mathbf{r}_i. \tag{6.17}$$

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.

6.3 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 [43]. 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 to address the following nonlinear navigation problem:

$$\dot{\mathbf{x}}(t) = \boldsymbol{f}[\mathbf{x}(t), \boldsymbol{\epsilon}(t), \boldsymbol{u}(t), t] + \boldsymbol{G}(t)\boldsymbol{w}(t) \tag{6.18}$$

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

$$\mathbf{z}_k^{[j]} = \mathbf{h}^{[j]}[\mathbf{x}(t), \boldsymbol{\epsilon}^{[j]}(t), \mathbf{u}(t), t, \mathbf{p}^{[j]}] + \boldsymbol{v}_k^{[j]}, \tag{6.19}$$

where $\mathbf{h}^{[j]}$ is the nonlinear measurement function for the j^{th} sensor, $\epsilon^{[\mathbf{j}]}$ is an $L \times 1$ subset of ϵ 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 $\boldsymbol{v}_k$ is a $Z \times 1$ discrete white noise process with covariance defined by matrix $\mathbf{R}_k^{[j]}$.

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

$$\mathbf{r}_k^{[j]} = \mathbf{z}_k^{[j]} - \mathbf{h}^{[j]}\left[\hat{\mathbf{x}}_k^-, \widehat{\boldsymbol{\epsilon}}_k^{[j]-}, \mathbf{u}_k, t_k, \hat{\mathbf{p}}_k^{[j]}\right], \tag{6.20}$$

where $\hat{\mathbf{x}}_k^-$, $\widehat{\boldsymbol{\epsilon}}_k^{[j]-}$, and $\hat{\mathbf{p}}_k^{[j]}$ are estimated quantities. Assuming white Gaussian noise, the measurement residual from 6.20 is expected to follow the distribution

$$\mathbf{r}_k^{[j]} \hookrightarrow \mathcal{N}\left(\mathbf{0}_{N\times 1}, \mathbf{S}_k^{[j]}\right), \tag{6.21}$$

$$\mathbf{S}_k^{[j]} = \mathbf{H}_k^{[j]}\mathbf{P}_k^-\mathbf{H}_k^{[j]^T} + \mathbf{R}_k^{[j]}, \tag{6.22}$$

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 (6.17) to monitor the statistical distribution of a user-defined monitoring period composed of recent Kalman pre-update residuals. A chi-square, χ^*, test statistic is used to detect excursions outside a user-defined threshold across the sampling period. Sensors in validation mode are excluded from impacting the main state estimates. Trusted sensors are directly 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 6.3.1.

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 $\boldsymbol{\epsilon}^{[j]}$ 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 $\boldsymbol{\epsilon}^{[j]}$), 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 **x**. The 'winning' sensor measurement model is selected based on which filter best matches the prescribed distribution (6.21) during the residual epoch. The sensor then enters validation mode. If the remodeling mode does not result in a new model selection, and 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 6.2 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.

One assumption that is not explicitly discussed in [43] is that ARMAS requires overlapping state observability [31] to detect anomalous sensor behavior. 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 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.

6.3.1 Sensor-Agnostic All-source Residual Monitoring (SAARM) .

As previously alluded, the monitoring mode in ARMAS was redesigned in 2019 with 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

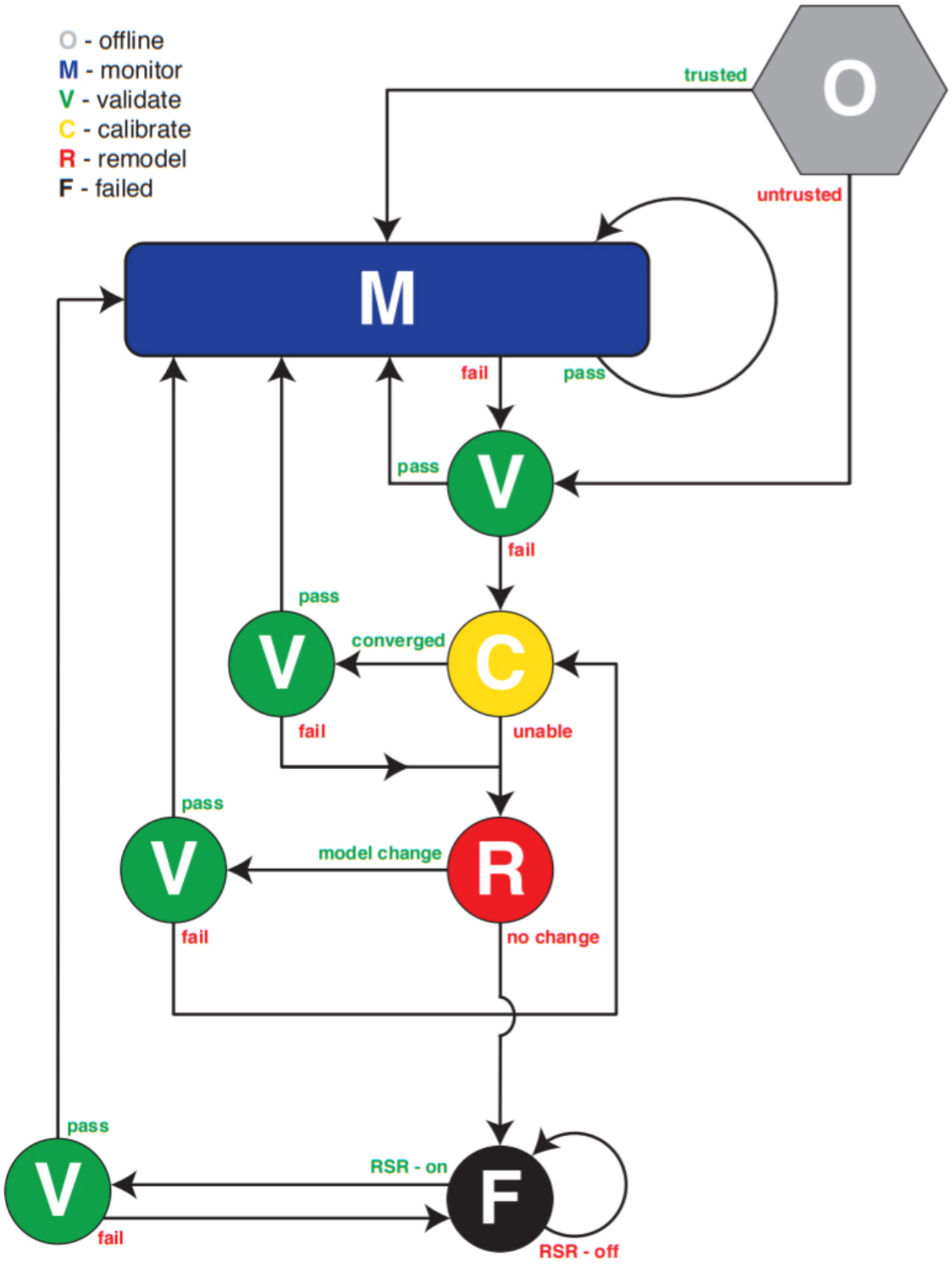

Figure 6.2: ARMAS Framework State Diagram [43]

sensor. The pluggable Bayesian filters provided by the SCORPION 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

$$\dot{\mathbf{x}}(t) = \boldsymbol{f}[\mathbf{x}(t), \boldsymbol{\epsilon}(t), \boldsymbol{u}(t), t] + \boldsymbol{G}(t)\boldsymbol{w}(t) \tag{6.23}$$

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

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

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

$$\mathbf{z}^{[i]}(t_k) = \mathbf{h}^{[i]}\big[\hat{\mathbf{x}}^{[j]}(t_k^-), \mathbf{u}(t_k), t_k\big] \tag{6.24}$$

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

$$\hat{\mathbf{z}}^{[i,j]}(t_k) = \mathbf{h}^{[i]}\big[\hat{\mathbf{x}}^{[j]}(t_k^-), \mathbf{u}(t_k), t_k\big], \tag{6.25}$$

where the estimated covariance matrix is defined by

$$\mathbf{P}_{\hat{\mathbf{z}}\hat{\mathbf{z}}}{}^{[i,j]}(t_k^-) = \mathbf{H}^{[i]}(t_k^-)\mathbf{P}_{\hat{\mathbf{x}}\hat{\mathbf{x}}}{}^{[j]}(t_k^-)\mathbf{H}^{[i]}(t_k^-)^T. \tag{6.26}$$

Using (6.25) and (6.26), the "pre-update residual" vector between sensor i and filter j, $\mathbf{r}^{[i,j]}$ and its covariance matrix, $\mathbf{P}_{rr}^{[i,j]}$ are defined as

$$\mathbf{r}^{[i,j]}(t_k) = \mathbf{z}^{[i]}(t_k) - \hat{\mathbf{z}}^{[i,j]}(t_k^-), \tag{6.27}$$

$$\mathbf{P}_{rr}^{[i,j]}(t_k) = \mathbf{R}^{[i]}(t_k) + \mathbf{P}_{\hat{z}\hat{z}}^{[i,j]}(t_k^-). \tag{6.28}$$

Fault detection relies on computing a moving average of recent residual-space test statistics formed by pre-update residual vectors from (6.27) and (6.28). 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

$$d^2 = (\mathbf{y} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{y} - \boldsymbol{\mu}), \tag{6.29}$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are the mean and covariance of a Z_i-dimensional Gaussian distribution. It is known that a sum of M independent d^2 distances follows a χ^* distribution with Z degrees of freedom [21] given by

$$\chi^* = \sum_{s=k}^{k+M} d^2(t_s), \tag{6.30}$$

$$d^2(t_k) = \mathbf{r}^T(t_k)[\mathbf{P}_{rr}(t_k)]^{-1}\mathbf{r}(t_k). \tag{6.31}$$

The set of pre-update residuals is known to be a zero-mean, white sequence [56]. The fault detection test for M pre-residuals is composed of the following hypotheses:

$$H_0 : \chi^*_{[i,j]} < \chi^2(1 - \alpha/2, M \times Z_i) \tag{6.32}$$

$$H_1 : \chi^*_{[i,j]} > \chi^2(1 - \alpha/2, M \times Z_i) \tag{6.33}$$

where α is the probablity of false alarm and M is the number of averaged pre-residual samples. 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). 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* [43].

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 6.5 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 6.2. 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:

$$J_N = \binom{I}{I-N} = \frac{I!}{N!(I-N)!} \tag{6.34}$$

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 SAARM subfilters!

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* [44]. 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.

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 χ^* statistical tests of pre-update measurement residuals. Fault exclusion is

based on a subfilter consensus approach which is tallied in a novel **T** matrix. A technique is introduced to provide resiliency to multiple simultaneous faults at the expense of factorial growth in the required quantity of concurrent estimation filters. 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.

6.3.2 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. Introduced in 2021, 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 6.2. 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.

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 6.3). 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 [44]. SOM allows the following extension of the previously stated fault exclusion axiom:

> *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* [44]. ***If the states of interest in each layer 2 subfilter are observable and stabilizable, then each layer 1 subfilter inherits these properties.***

Assuming no more than one simultaneous sensor failure, this means that the physical region encompassed by the position covariance estimates of all layer 1 subfilters contains the true navigation state within the statistical significance of the fault detection tests.

SOM records and monitors 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

$$O_{k,pos}(n) = \begin{cases} 1, & \text{if } tr(\mathbf{P}^{[n]}_{pos}(t_k^+)) > tr(\mathbf{P}^{[n]}_{pos}(t_{k-M}^+))\beta \\ 1, & \text{if } tr(\mathbf{P}^{[n]}_{pos}(t_k^+)) > tr(\mathbf{P}_{pos,max}) \\ 0, & \text{otherwise} \end{cases} \tag{6.35}$$

where $\mathbf{P}^{[n]}_{pos}(t_k^+)$ is the most recent post-update position covariance matrix for layer 2 subfilter n, $\mathbf{P}^{[n]}_{pos}(t_{k-M}^+)$ is the post-update position covariance matrix for layer 2 subfilter n exactly M samples prior to t_k $\mathbf{P}_{pos,max}$ is a user-defined limit for maximum steady-state position state estimate covariance, and $\beta \in [1, \infty)$ is the state estimate covariance transient growth threshold. 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 $tr(\mathbf{P}^{[n]}_{pos}(t_k^+))$ converges, then the individual

position estimate variances also converge. When applying (6.35), 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 (6.36). The newly added sensor will directly enter monitoring mode if it is considered 'trusted' or must pass through sensor validation if 'untrusted'.

$$SOM_{Collaborate} = \begin{cases} true, & \text{if } \sum_{n=1}^{N} O_k(n) > 0 \\ false, & \text{otherwise} \end{cases} \tag{6.36}$$

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. The results of (6.36) are ignored until the newly requested sensor completes an ARMAS monitoring epoch. If post update variance stability is regained, then (6.36) will set to 0 for each stable layer 2 subfilter and the observability warning is rescinded. This method flags the presence of an information deficiency with respect to the estimated states of interest in real-time. Due to potentially variable navigation state information available from all-source navigation sensors, it is critical that ARMAS is always "aware" of when it lacks enough information to maintain consistency.

6.3.3 Estimator Credibility Analysis.

Estimator performance is always dependent on available measurements [56]. 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 $\widetilde{\boldsymbol{x}}$ of state estimate $\widehat{\boldsymbol{x}}$ at time i is shown by (6.37). The MSE of $\widetilde{\boldsymbol{x}}$ is shown by (6.38).

$$\widetilde{\boldsymbol{x}}_i = \boldsymbol{x}_i - \widehat{\boldsymbol{x}}_i \tag{6.37}$$

$$MSE = \frac{1}{n}\widetilde{\boldsymbol{x}}_i^T\widetilde{\boldsymbol{x}}_i \tag{6.38}$$

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 [18]. NEES is calculated according to Eq. 6.39.

$$\gamma_i = (x_i - \widehat{x_i})^T \boldsymbol{P}_i^{-1} (x_i - \widehat{x_i}) \tag{6.39}$$

For an m-dimensional state estimate with n total measurements, the Average NEES is shown by Eq. 2.56.

$$\overline{\gamma} = \frac{1}{nm} \sum_{i=1}^{n} \gamma_i \tag{6.40}$$

NEES represents the square of the distance between the state estimate $\widehat{\boldsymbol{x}}$ and the truth $\boldsymbol{x}$, normalized by the error covariance $\boldsymbol{P}$. Average NEES can be interpreted as a uni-dimensional average of this squared distance. Average NEES quantifies estimator credibility by assessing the accuracy of employed assumptions [18]. 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).

6.4 Methods for Managing Cross-Correlated Information

For the purposes of this discussion, 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 [41]. Distributed networks are scalable and resilient to changes in topology but are particularly

vulnerable to this problem. If dependencies are not properly managed, the estimator can become inconsistent by overestimating available sensor information and eventually diverge.

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.

6.4.0.1 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 [41][40] 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 $\mathbf{P}_{ab}$ 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*.

6.4.0.2 Interleaved Update (IU) Algorithm.

The IU algorithm [7] 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 (6.41).

$$\mathbf{P_{i,j}}(t) = \left[\begin{array}{c|c} P_i(t) & 0 \\ \hline 0 & P_j(t) \end{array}\right] \tag{6.41}$$

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 $argmin[trace(\mathbf{P}_i^q)]$ 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 [7].

Exponential scalability is highly undesirable, especially for multi-filter residual monitoring algorithms like ARMAS [70]. 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.

6.4.0.3 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 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 is already propagating a subfilter which never contained 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 (Figure 6.4). 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 [25]. In other words, a single fault detection and exclusion 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. Lastly, this approach can be used to prioritize high-fidelty donors over less desirable ones.

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 6.5 shows the structural similarities **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.

6.5 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.

6.6 Simulation

Consider a 3-D 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

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \dot{\mathbf{x}}_p(t) \\ \dot{\mathbf{x}}_v(t) \\ \dot{\mathbf{x}}_a(t) \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{x}}_v(t) \\ \dot{\mathbf{x}}_a(t) \\ -\frac{1}{\tau_a}\dot{\mathbf{x}}_v(t) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \mathbf{w(t)} \end{bmatrix} \tag{6.42}$$

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+τ)^T] = $\mathbf{Q}\delta(\tau)$ and

$$\mathbf{Q} = (1.0 \times 10^{-2})^2 \mathbf{I}_{3\times 3}\ (m^2/s^4). \tag{6.43}$$

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

$$\mathbf{P_n}(0) = diag(\begin{bmatrix} 20^2 & 20^2 & 20^2 & 10^2 & 10^2 & 10^2 & 0.01^2 & 0.01^2 & 0.01^2 & 8000^2 \end{bmatrix}). \tag{6.44}$$

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 (6.45)

$$\rho_i = \sqrt{(X_{SV,i} - X_u) + (Y_{SV,i} - Y_u) + (Z_{SV,i} - Z_u)} + b_u \tag{6.45}$$

where ρ_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

$$R_j = \sqrt{(X_{wng,j} - X_r) + (Y_{wng,j} - Y_r) + (Z_{wng,j} - Z_r)}. \tag{6.46}$$

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 (6.46) to form the measurement update. The main solution from the user's collaborative instance of ARMAS-SOM provides the estimated user location and covariance. The wingman's proprioceptive instance of ARMAS-SOM is used to provide the estimated wingman location and associated position covariance. The ranging measurement covariance is $\mathbf{R}_{WING} = 10^2$ m^2. Collaborative updates are performed only in the collaborative instance of ARMAS-SOM for each recipient. Clock synchronization is assumed for collaborative measurements. Vehicles are equipped with simulated alternative visual navigation sensors which provide noisy velocity updates with $\mathbf{R}_{VISNAV} = (10^2)\mathbf{I}_{\mathbf{3\times 3}}$ (m/s)^2. All sensors provide measurement updates at 1Hz. SOM a posteriori state estimate covariance parameters are set to $\beta = 5$ and $\mathbf{P}_{j,max} = (25\text{m})^2 \times 3 = 6889\text{m}^2$. The vehicles travel at approximately 75 meters/second in the $+Y$ direction in a local tangential reference frame for a total of 27 minutes (1620 seconds) of run time.

The simulation is divided into 4 separate regions (Figure 6.6):

1. Permissive (0-5km)
2. L1 Jamming (5-50km)
3. L1 Jamming + Single Pseudorange 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 $1 \times VISNAV$ 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 $1 \times VISNAV$ 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 6.4.0.3 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 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 6.7). 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.

6.7 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 6.7. A comparison of the GPZ dimensions for both the proprioceptive and collaborative ARMAS-SOM instances is shown in Figure 6.8. 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 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.

Figure 6.9 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 6.1. Stingy collaboration resulted in a 11.3% mean reduction in RSS error across all vehicles for the duration of the simulation. Average NEES for both the proprioceptive and collaborative instances across all vehicles were 0.92 and 0.90, respectively. 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.

6.8 Chapter Summary and Future Work

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

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 ANEES	0.94	1.16	0.74	0.98	0.78	0.92
"Stingy" Collaborative ANEES	0.88	1.04	0.69	1.10	0.81	0.90

Table 6.1: Comparison of Proprioceptive and Collaborative ARMAS-SOM Instances

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.

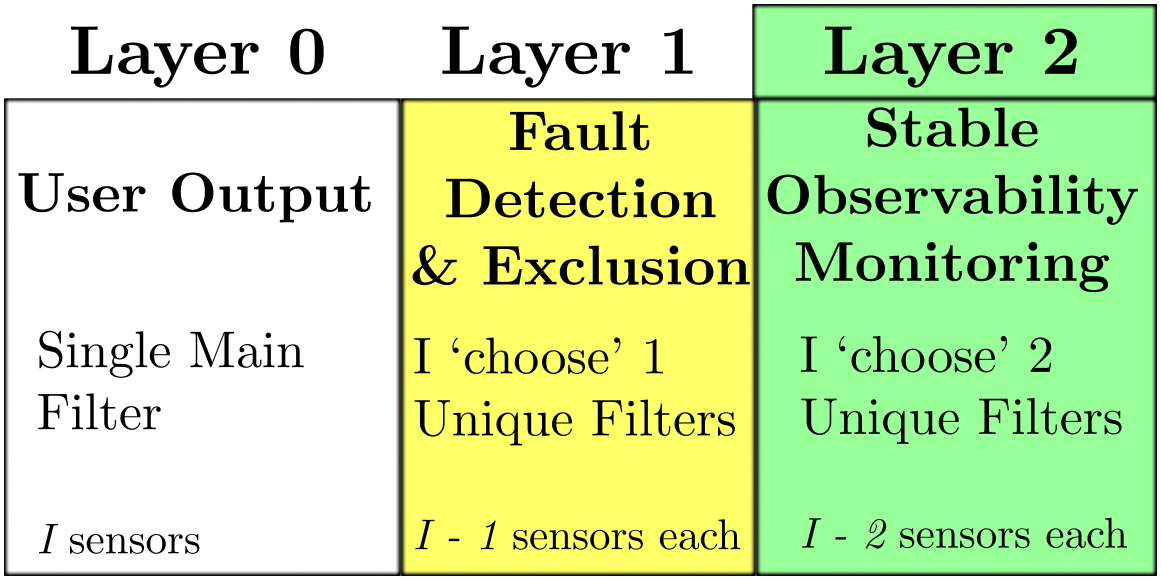

Figure 6.3: ARMAS Monitoring Mode with User Output Layer 0, Fault Detection and Exclusion Sub-Layer 1, and Observability Sub-Layer 2 for Stable Observability Monitoring (SOM)

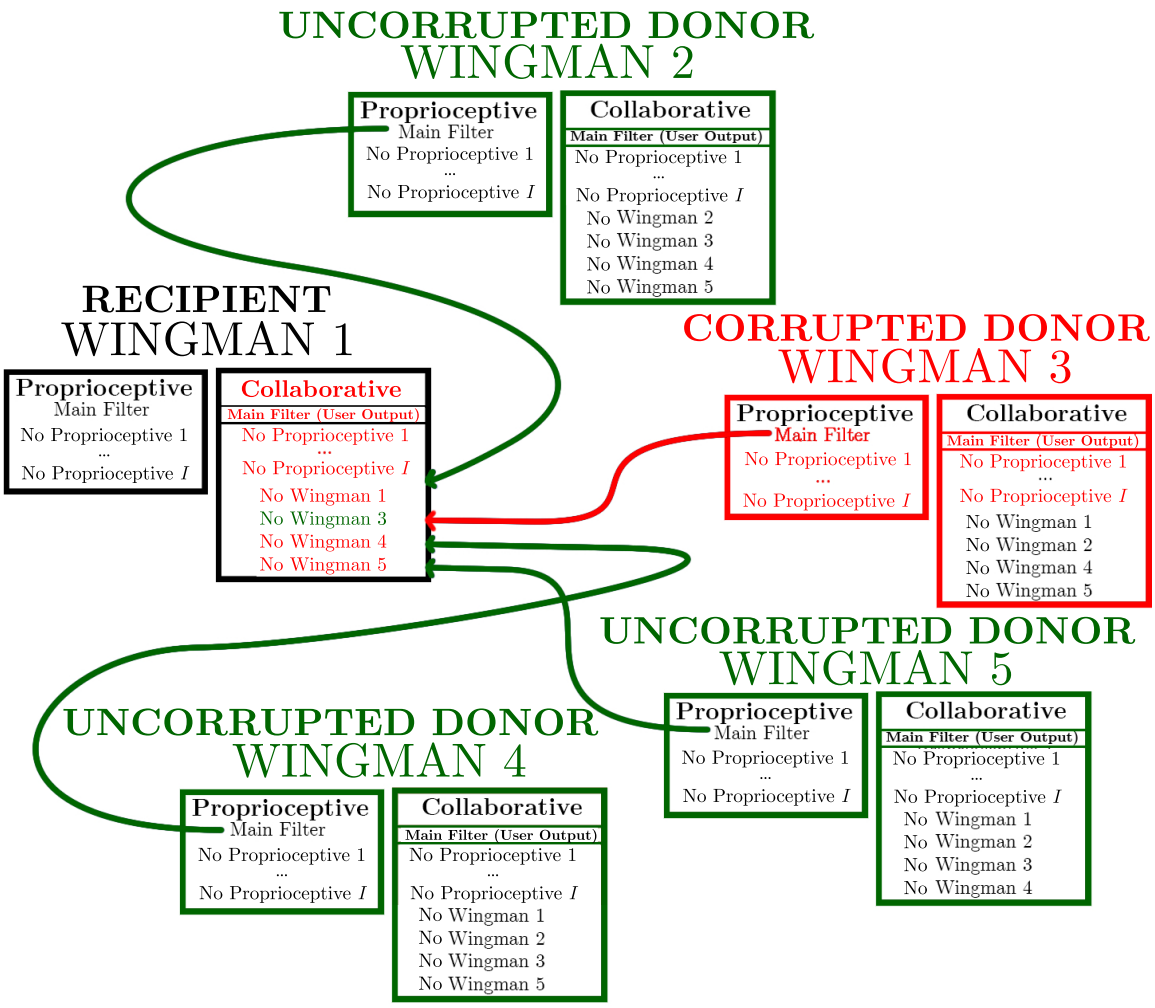

Figure 6.4: Stingy Collaboration Example: Wingman 1 receives corrupted collaborative donor information from Wingman 3. A single fault detection and exclusion action is required to remove all historical contributions from Wingman 3.

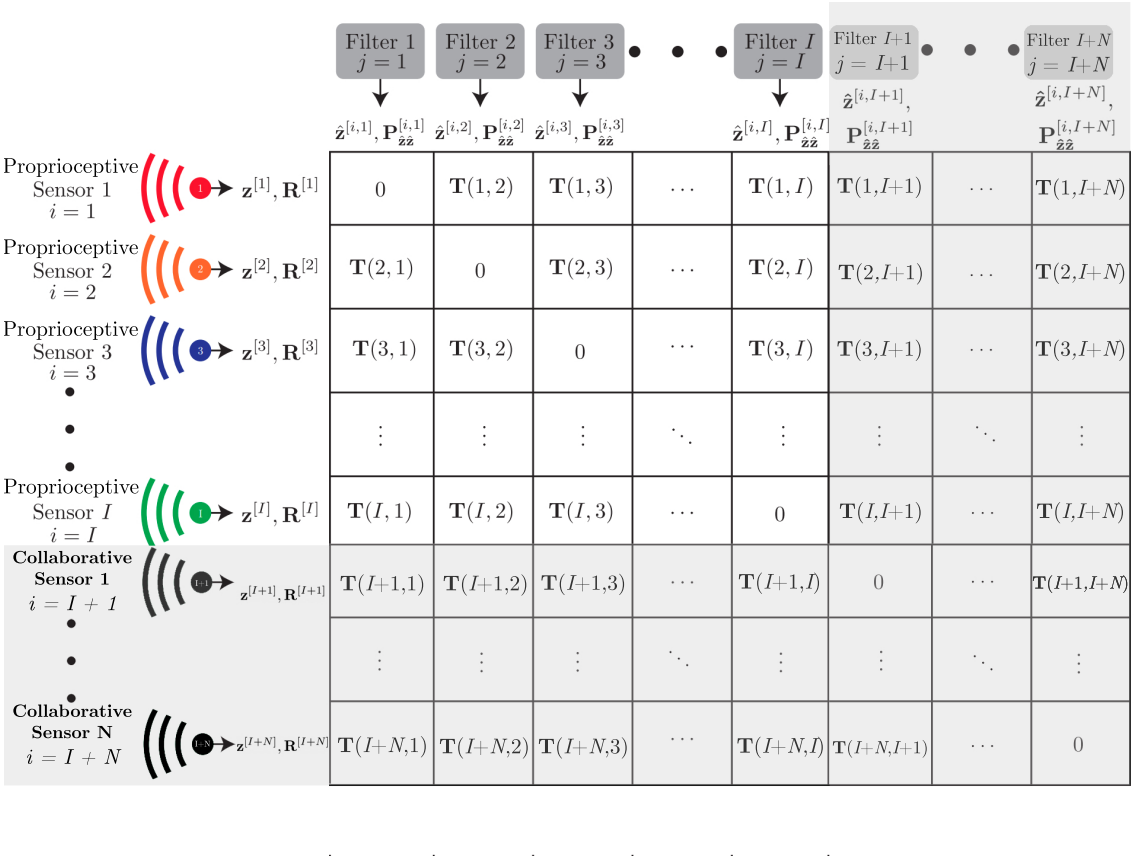

Figure 6.5: Collaborative ARMAS **T** matrix: Collaborative information (gray) is used to augment proprioceptive information (white). This forms the **T** matrix structure for the collaborative all-source framework which is fused into a single main filter for user output. A parallel proprioceptive instance is maintained separately.

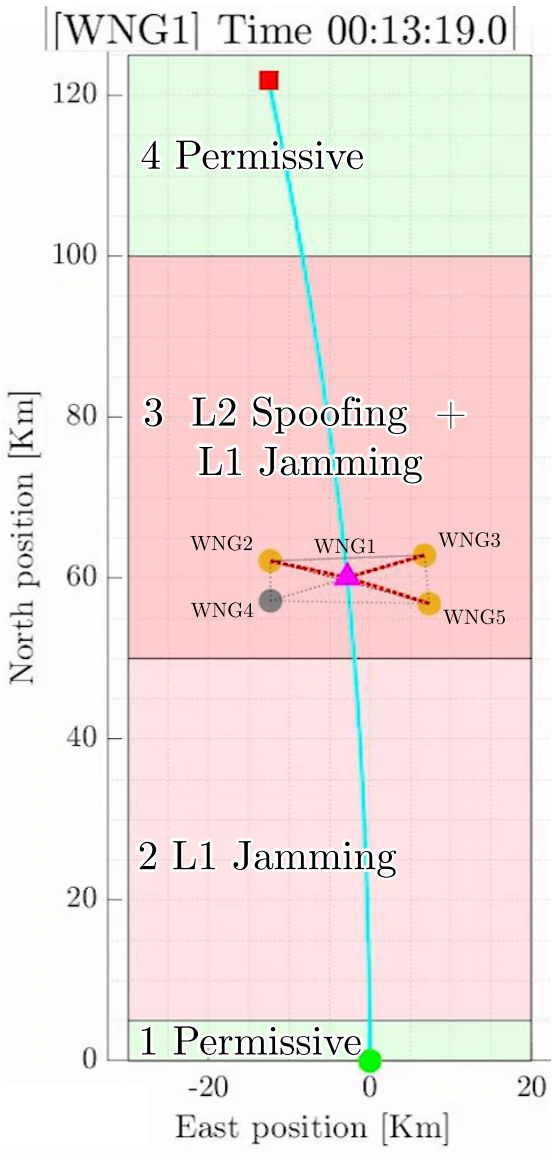

Figure 6.6: Overhead view of GNSS Regions with WNG1 in center of formation at T = 13m 19s (799s). Wingman locations are: WNG2 (Upper Left), WNG3 (Upper Right), WNG4 (Lower Left), WNG5 (Lower Right)

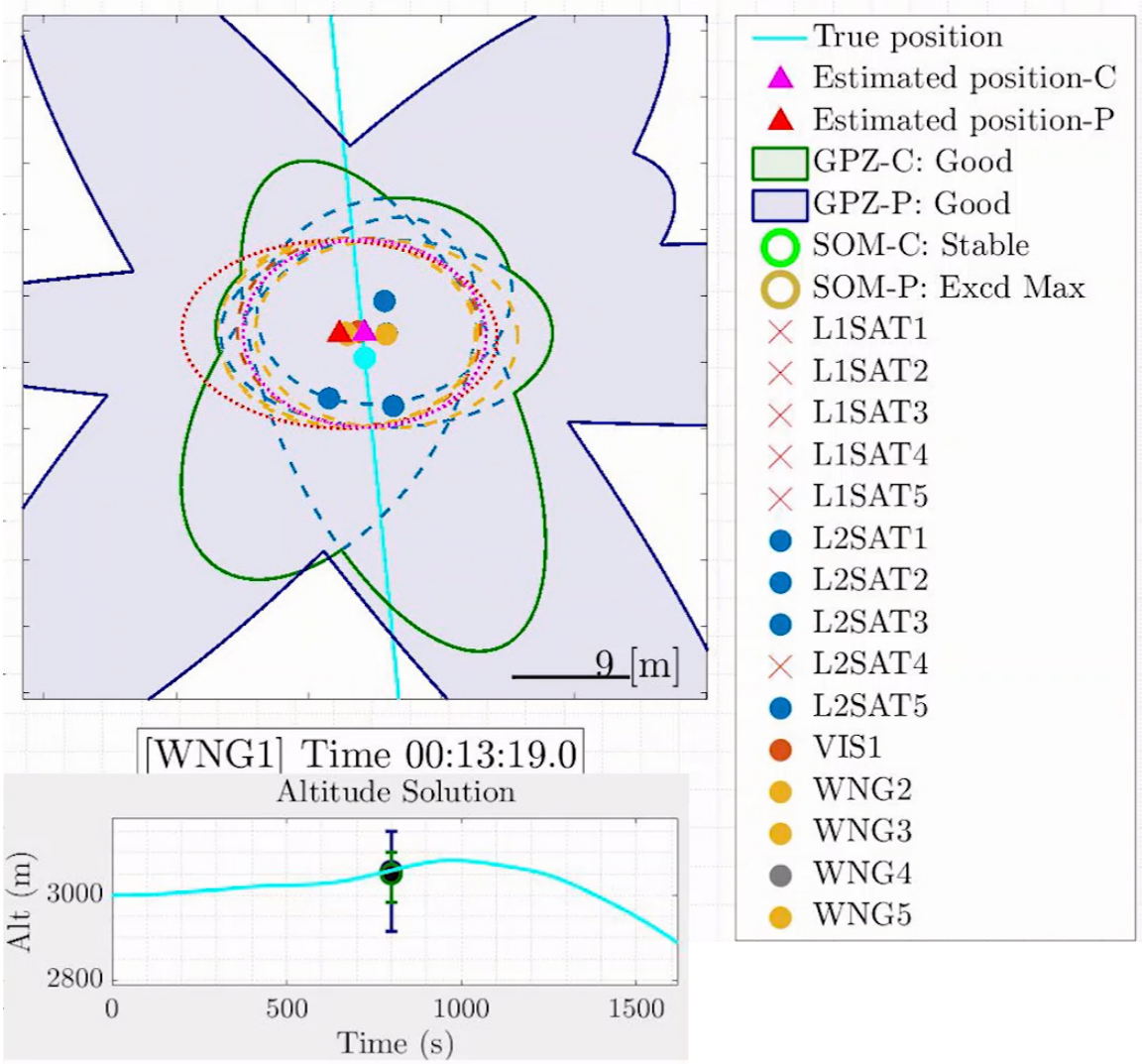

Figure 6.7: Combined Overhead and Profile View of WNG1 Split Collaborative "GPZ-C" and Proprioceptive "GPZ-P" ARMAS-SOM Instances with individual Layer 1 Subfilter Solutions at T = 13m 19s (T = 799s) inside the L1 Jamming + L2 Spoofing Region 3

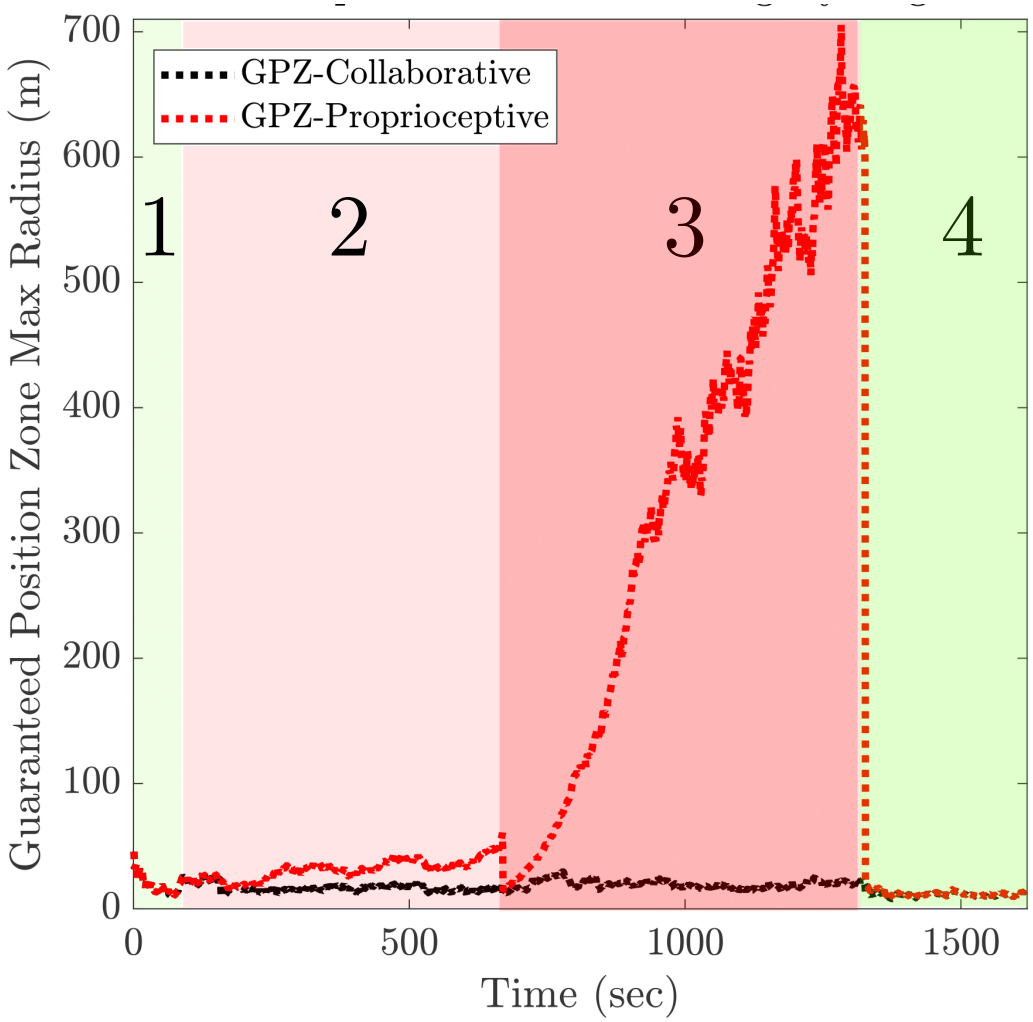

Figure 6.8: Size Comparison of Split "Stingy" Collaborative and Proprioceptive All-source Guaranteed Position Zones for WNG1 in GNSS Regions 1-4

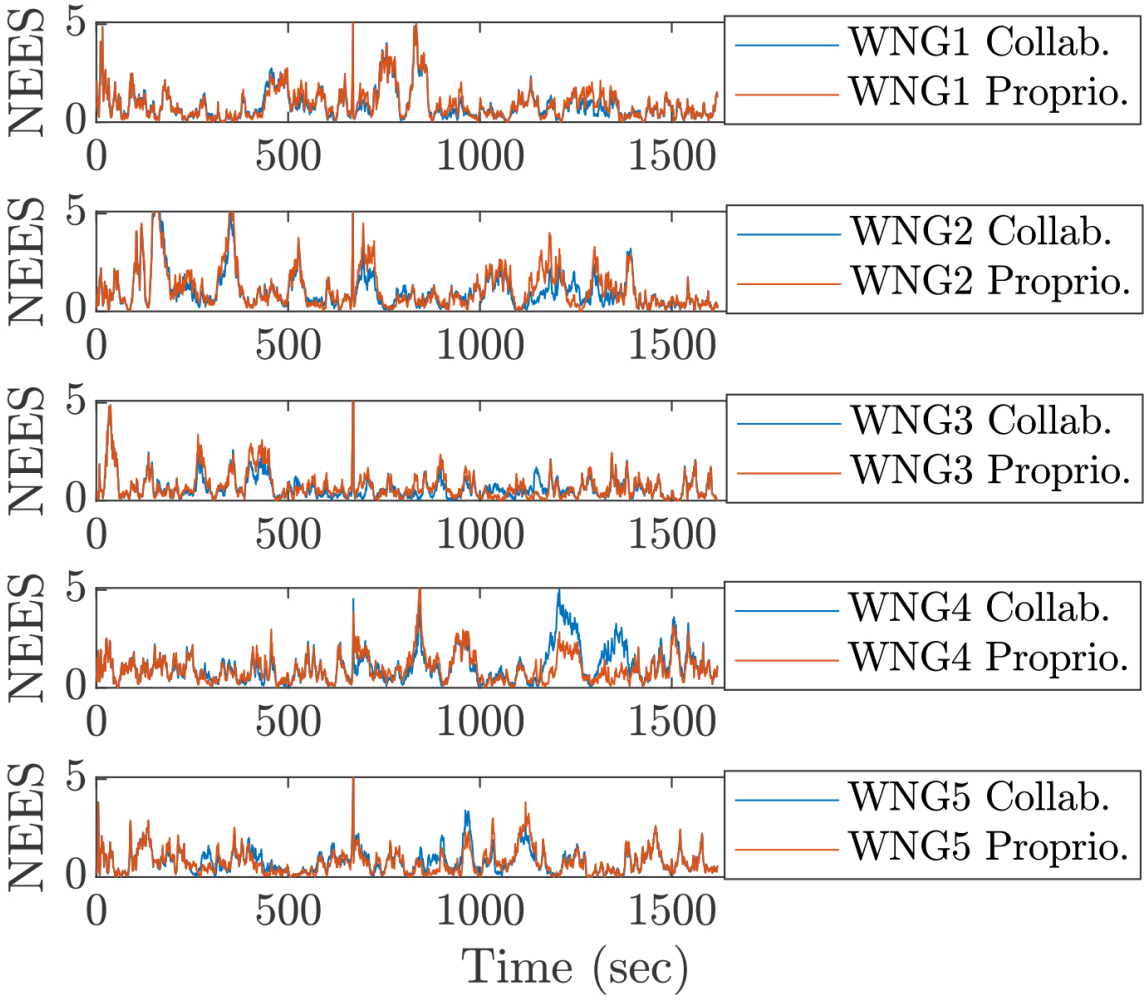

Figure 6.9: Estimator Credibility Comparison for Split Collaborative and Proprioceptive Instances of ARMAS-SOM for 5 Vehicles

VII. Summary and Conclusions

As we learned, the United States' need for resilient PNT is congruous across agencies responsible for domestic infrastructure (e.g. DHS, CISA) and national defense (e.g. DoD, USAF, USA, USN). Executive Order 13905 "Strengthening National Resilience Through Responsible Use of Positioning, Navigation, and Timing Services" [1] made this a clear national priority. The 2030 United States Air Force Science and Technology Strategy [36] expressed a strategic challenge to obtain a mix of capable, trustworthy, autonomous systems that can perform as part of a collaborative system to provide synergistic, multi-domain effects. With this in mind, this research presented methods for collaborative all-source navigation and real-time autonomous guidance for single and multi-vehicle cooperative navigation in GNSS-denied environments. The unifying goal of these pursuits is to increase the reliability and trustworthiness of collaborative PNT for critical domestic infrastructure and defense capabilities.

Chapter 3 contributed a real-time heuristic method of trajectory generation for a single aircraft to minimize relative positioning error in a wireless sensor network composed of a simulated formation of maneuvering aircraft. This control scheme, SAS, was based on a compact, scalable particle swarm to locally optimize a pluggable cost function in real-time. Chapter 4 presented a distributed real-time swarm control scheme with the introduction of the SCANS algorithm. SCANS demonstrated real-time swarm control for importance-weighted navigation support. This effort built on the concept of greedy local optimization of globally convex cost surfaces (proven in Chapter 3) to demonstrate consensus-like behavior.

Chapter 5 contributed a definition and assessment of a position state observability margin which provided intrinsic navigation state awareness missing from ARMAS. The combined framework resulting from this contribution became known as ARMAS-

SOM and enabled the demonstration of resilient all-source navigation through proactive augmentation of navigation state observability. The intrinsic information awareness also indicated when it was safe to break collaborative information links and transition back to autonomy. Chapter 6 introduced a novel "stingy collaboration" framework which built on the resilient fault detection and exclusion, online calibration, and remodeling of ARMAS-SOM. Stingy collaboration demonstrated resilient all-source navigation with integrity in a complex collaborative 3-D simulation. In summary, we offer the following guidelines gleaned from this collaborative all-source navigation research:

- The key to navigation 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.

Bibliography

[1] "Executive Order 13905: Strengthening National Resilience Through Responsible Use of Positioning, Navigation, and Timing Services". *Office of the President of the United States*, 2020.

[2] Amirsadri, Ashkan, Adrian N. Bishop, Jonghyuk Kim, Jochen Trumpf, and Lars Petersson. "Consistency analysis for data fusion: Determining when the unknown correlation can be ignored". *2013 International Conference on Control, Automation and Information Sciences, ICCAIS 2013*, 97–102, 2013.

[3] Angjelichinoski, Marko, Student Member, Daniel Denkovski, Student Member, Vladimir Atanasovski, Senior Member, Liljana Gavrilovska, and Senior Member. "Cramér – Rao Lower Bounds of RSS-Based Localization With Anchor Position Uncertainty". *IEEE Transactions on Information Theory*, 61(5):2807–2834, 2015.

[4] Appleget, Andy, Robert C. Leishman, and Jonathon S. Gipson. "Evaluation of Sensor-Agnostic All-Source Residual Monitoring for Navigation". *IEEE/ION PLANS*, 2021.

[5] Ash, Joshua N, Spyros Kyperountas, Alfred O Hero Iii, Randolph L Moses, and Neiyer S Correal. "Locating the nodes: cooperative localization in wireless sensor networks". *IEEE Signal Processing Magazine*, 22(July):54–69, 2005.

[6] Ash, Joshua N. and Randolph L. Moses. "On the relative and absolute positioning errors in self-localization systems". *IEEE Transactions on Signal Processing*, 56(11):5668–5679, 2008. ISSN 1053587X.

[7] Bahr, Alexander, Matthew R. Walter, and John J. Leonard. "Consistent cooperative localization". *Proceedings - IEEE International Conference on Robotics and Automation*, 3415–3422, 2009. ISSN 10504729.

[8] Blanch, Juan, Todd Walker, Per Enge, Young Lee, Boris Pervan, Markus Rippl, Alex Spletter, and Victoria Kropp. "Baseline advanced RAIM user algorithm and possible improvements". *IEEE Transactions on Aerospace and Electronic Systems*, 51(1):713–732, 2015. ISSN 00189251.

[9] Brink, Kevin M. "Partial-update Schmidt-Kalman filter". *Journal of Guidance, Control, and Dynamics*, 40(9):2214–2228, 2017. ISSN 15333884. URL https://doi.org/10.2514/1.G002808.

[10] Brown, Alison K. "GPS/INS uses Low-Cost MEMS IMU". (September):3–10, 2005.

[11] Bullard, Randall, John Colombi, and G. Richard Freeman. "Global positioning system: A case study focused on systems engineering". *Proceedings of 19th International Conference on Systems Engineering, ICSEng 2008*, 3–7, 2008.

[12] Canciani, Aaron. "Absolute Positioning Using the Earth's Magnetic Anomaly Field, Dissertation, Air Force Institute of Technology". *ION PLANS 2015*, 1(March), 2015.

[13] Canciani, Aaron and John Raquet. "Absolute positioning using the Earth's magnetic anomaly field". *Navigation*, 63(2):111–126, 2016.

[14] Canciani, Aaron and John Raquet. "Airborne Magnetic Anomaly Navigation". *IEEE Transactions on Aerospace and Electronic Systems*, 53(1):67–80, 2017. ISSN 00189251.

[15] Cappel, D. "Optimal Observer Maneuver for Bearings-Only Tracking". 34(3), 1998.

[16] Carvalho, H., P. Del Moral, A. Monin, and G. Salut. "Optimal nonlinear filtering in GPS/INS integration". *IEEE Transactions on Aerospace and Electronic Systems*, 33(3):835–850, 1997. ISSN 00189251.

[17] Cavaleri, Antonio, Beatrice Motella, Marco Pini, and Maurizio Fantino. "Detection of spoofed GPS signals at code and carrier tracking level". *Programme and Abstract Book - 5th ESA Workshop on Satellite Navigation Technologies and European Workshop on GNSS Signals and Signal Processing, NAVITEC 2010*, 1–6, 2010.

[18] Chen, Zhaozhong, Christoffer Heckman, Simon Julier, and Nisar Ahmed. "Weak in the NEES?: Auto-Tuning Kalman Filters with Bayesian Optimization". *2018 21st International Conference on Information Fusion, FUSION 2018*, 1072–1079, 2018.

[19] Chen, Zhe. "Local Observability and Its Application to Multiple Measurement Estimation". *IEEE Transactions on Industrial Electronics*, 38(6):491–496, 1991. ISSN 15579948.

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

[21] De Maesschalck, Roy, Delphine Jouan-Rimbaud, and Désiré L Massart. "The mahalanobis distance". *Chemometrics and intelligent laboratory systems*, 50(1):1–18, 2000.

[22] Faragher, Ramsey and Robert Harle. "Location fingerprinting with bluetooth low energy beacons". *IEEE Journal on Selected Areas in Communications*, 33(11):2418–2428, 2015. ISSN 07338716.

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

[24] Gipson, Jonathon S. *Air-to-air missile enhanced scoring with Kalman smoothing*. Master's thesis, Air Force Institute of Technology, March 2012.

[25] Gipson, Jonathon S and Robert C Leishman. "Resilience for Multi-filter All-source Navigation Framework with Integrity". *AFIT Faculty Preprint*, 2021.

[26] Gipson, Jonathon S., Robert C. Leishman, and Christine M. Schubert Kabban. "Swarm Control for Autonomous Navigation Support". *2020 International Conference on Unmanned Aircraft Systems, ICUAS 2020*, 446–455, 2020.

[27] Gipson, Jonathon S, Robert C. Schubert-Kabban Christine M., Leishman, and Juan Jurado. "Real-time Trajectory Optimization for Collaborative Self-Localization in Random Aircraft Formations". *IEEE/ION Plans*, 2020.

[28] Giremus, Audrey, Jean Yves Tourneret, and Petar M. Djuric. "An improved regularized particle filter for GPS/INS integration". *IEEE Workshop on Signal Processing Advances in Wireless Communications, SPAWC*, 2005:1013–1017, 2005.

[29] Givens, Matthew, Calvin Coopmans, and Randall Christensen. "An estimation-domain approach to MEMS Multi-IMU fusion for sUAS". *2019 International Conference on Unmanned Aircraft Systems, ICUAS 2019*, 1048–1053, 2019.

[30] Hag Elamin, Khalid Abd El Mageed and Mirghani Fath Elrahman Taha. "On the steady-state error covariance matrix of Kalman filtering with intermittent observations in the presence of correlated noises at the same time". *Proceedings - 2013 International Conference on Computer, Electrical and Electronics Engineering: 'Research Makes a Difference', ICCEEE 2013*, 15–22, 2013.

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

[32] Helferty, James P, David R Mudgett, and John E Dzielski. "Trajectory Optimization for Minimum Range Error in Bearings-only Source Localization ". 1993.

[33] Hermann, Robert and Arthur J. Krener. "Nonlinear Controllability and Observability". *IEEE Transactions on Automatic Control*, 22(5):728–740, 1977. ISSN 15582523.

[34] Holme, R. "The Magnetic Field of the Earth's Lithosphere R. A. Langel W. J. Hinze, Cambridge University Press, Cambridge, 1998, 429 pp, ISBN 0-521-47333-0, Hardback, £60 (US 100)". *Geophysical Journal International*, 137(3):909–909, 1999.

[35] Hong, Sinpyo, Ho Hwan Chun, Sun Hong Kwon, and Man Hyung Lee. "Observability measures and their application to GPS/INS". *IEEE Transactions on Vehicular Technology*, 57(1):97–106, 2008. ISSN 00189545.

[36] HQ, United States Air Force. "Science and Technology Strategy, 2030 and Beyond". *TAPPI Journal*, 18(3), 2019.

[37] Huang, Grant, Brian K. Taylor, and David Akopian. "A Low-Cost Approach of Magnetic Field-Based Location Validation for Global Navigation Satellite Systems". *IEEE Transactions on Instrumentation and Measurement*, 68(12):4937–4944, 2019. ISSN 15579662.

[38] Hurley, Michael B. "An information theoretic justification for covariance intersection and its generalization". *Proceedings of the 5th International Conference on Information Fusion, FUSION 2002*, 1:505–511, 2002.

[39] Jia, Tao, Student Member, R Michael Buehrer, and Senior Member. "Collaborative Position Location". *IEEE Transactions on Wireless Communications*, 9(1):374–383, 2010.

[40] Julier, S. J. "Fusion without independence". *IET Seminar Digest*, 2008(12273):1–4, 2008.

[41] Julier, Simon J. and Jeffrey K. Uhlmann. "Non-divergent estimation algorithm in the presence of unknown correlations". *Proceedings of the American Control Conference*, 4:2369–2373, 1997. ISSN 07431619.

[42] Jurado, Juan, John Raquet, and Christine M Schubert Kabban. "Single-Filter Finite Fault Detection and Exclusion Methodology for Real-time Validation of Plug-and-play Sensors". *IEEE Transactions on Aerospace and Electronic Systems*, 2020.

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

[44] Jurado, Juan, John F. Raquet, Christine M. Schubert-Kabban, and Jonathon S Gipson. "Residual-Based Multi-Filter Methodology for All-Source Fault Detection, Exclusion, and Performance Monitoring". *ION*, 2019.

[45] Kalman, R. E. "A New Approach to Linear Filtering and Prediction Problems". *Journal of Basic Engineering*, 82(1):35, 1960. URL https://doi.org/10.1115%2F1.3662552.

[46] Kauffman, Kyle, Daniel Marietta, John Raquet, Daniel Carson, Robert C Leishman, Aaron Canciani, Adam Schofield, and Michael Caporellie. "Scorpion : A Modular Sensor Fusion Approach for Complementary Navigation Sensors". *ION/IEEE PLANS*. IEEE, Portland, OR, 2020.

[47] Kauffman, Kyle, Daniel Marietta, John Raquet, Daniel Carson, Robert C. Leishman, Aaron Canciani, Adam Schofield, and Michael Caporellie. "Scorpion: A Modular Sensor Fusion Approach for Complementary Navigation Sensors". *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*. Portland, OR, April 2020.

[48] LeCadre, J. P. "Properties of Estimability Criteria for Target Motion Analysis". 145(2), 1998.

[49] Leonardos, Spyridon and Kostas Daniilidis. “A game-theoretic approach to Robust Fusion and Kalman filtering under unknown correlations”. *Proceedings of the American Control Conference*, (1):2568–2573, 2017. ISSN 07431619.

[50] Li, Maodeng, Dayi Wang, and Xiangyu Huang. “Study on the observability analysis based on the trace of error covariance matrix for spacecraft autonomous navigation”. *IEEE International Conference on Control and Automation, ICCA*, 95–98, 2013. ISSN 19483449.

[51] Li, Wei, Zhen Yang, and Haifeng Hu. “A new variant of sum-product algorithm for sensor self-localization in wireless networks”. *2013 IEEE/CIC International Conference on Communications in China, ICCC 2013*, (Iccc):628–632, 2013.

[52] Li, X Rong, Zhanlue Zhao, and Vesselin P Jilkov. “Practical measures and test for credibility of an estimator”. *Proc. Workshop on Estimation, Tracking, and . . .*, 481—-495, 2001. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.142.6049{&}rep=rep1{&}type=pdf.

[53] Liebe, Carl Christian. “Accuracy performance of star trackers - A tutorial”. *IEEE Transactions on Aerospace and Electronic Systems*, 38(2):587–599, 2002. ISSN 00189251.

[54] Liu, Mingyong, Wenbai Li, Bingxian Mu, and Fuqiang Liu. “Cooperative navigation for multiple AUVs based on relative range measurements with a single leader”. *Proceedings - 2010 IEEE International Conference on Intelligent Computing and Intelligent Systems, ICIS 2010*, 2:762–766, 2010.

[55] Martin, Richard K. “Based Positioning Performance”. *IEEE Signal Processing Letters*, 18(12):741–744, 2011.

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

[57] Niu, Ruixin, Aditya Vempaty, and Pramod K. Varshney. “Received-Signal-Strength-Based Localization in Wireless Sensor Networks”. *Proceedings of the IEEE*, 106(7):1166–1182, 2018. ISSN 00189219.

[58] O’Hanlon, Brady W., Mark L. Psiaki, Jahshan A. Bhatti, Daniel P. Shepard, and Todd E. Humphreys. “Real-time GPS spoofing detection via correlation of encrypted signals”. *Navigation, Journal of the Institute of Navigation*, 60(4):267–278, 2013. ISSN 00281522.

[59] Powel, Nathan D. and Kristi A. Morgansen. “Empirical Observability Gramian for Stochastic Observability of Nonlinear Systems”. *arXiv*, (Cdc):6342–6348, 2020. ISSN 23318422.

[60] Proletarsky, Andrey V., Konstantin A. Neusypin, Kai Shen, Maria S. Selezneva, and Vic Grout. "Development and analysis of the numerical criterion for the degree of observability of state variables in nonlinear systems". *2017 Internet Technologies and Applications, ITA 2017 - Proceedings of the 7th International Conference*, (2):150–154, 2017.

[61] Psiaki, Mark L and Todd E Humphreys. "GPS Lies". *IEEE Spectrum*, 2016.

[62] Quartararo, John D. and Steven E. Langel. "Detecting slowly accumulating faults using a bank of cumulative innovations monitors in kalman filters". *Proceedings of the 33rd International Technical Meeting of the Satellite Division of the Institute of Navigation, ION GNSS+ 2020*, (20):2367–2381, 2020.

[63] Raquet, John and Richard K. Martin. "NON-GNSS RADIO FREQUENCY NAVIGATION, The Air Force Institute of Technology Dept . of ECE". *Signals*, 5308–5311, 2008.

[64] Rhee, Ihnsoek, Mamoun F. Abdel-Hafez, and Jason L. Speyer. "Observability of an integrated GPS/INS during maneuvers". *IEEE Transactions on Aerospace and Electronic Systems*, 40(2):526–535, 2004. ISSN 00189251.

[65] Rouaud, Mathieu. "Probability, statistics and estimation". *Propagation of uncertainties*, 191, 2013. URL http://www.incertitudes.fr/book.pdf.

[66] Roumeliotis, Stergios I. and George A. Bekey. "Distributed multirobot localization". *IEEE Transactions on Robotics and Automation*, 18(5):781–795, 2002. ISSN 1042296X.

[67] Roy, Prateep, Arben Çela, and Yskandar Hamam. "On the relation of FIM and controllability gramian". *Proceedings - 2009 IEEE International Symposium on Industrial Embedded Systems, SIES 2009*, 37–41, 2009.

[68] Schmitt, John J. "Some Considerations in the Design of the Guidance and Control System for Discoverer". *IRE Transactions on Military Electronics*, MIL-3(4):184–185, 1959. ISSN 00962511.

[69] Schuler, M. "The Perturbation of Pendulum and Gyroscope Instruments by Acceleration of the Vehicle". *Physik. Zeitschr.*, 24:344, 1923. URL http://scholar.google.com/scholar?hl=en{&}btnG=Search{&}q=intitle:The+perturbation+of+pendulum+and+gyroscope+instruments+by+acceleration+of+the+vehicle{#}2.

[70] Sepulveda, Louis, Robert Leishman, Kyle Kauffman, and Jonathon Gipson. "Optimizing a Bank of Kalman Filters for Navigation Integrity using Efficient Software Design." *Proceedings of the ION GNSS+*, 2021.

[71] Shames, Iman, Baris Fidan, Brian D.O. Anderson, and Hatem Hmam. "Cooperative self-localization of mobile agents". *IEEE Transactions on Aerospace and Electronic Systems*, 47(3):1926–1947, 2011. ISSN 00189251.

[72] Systems, Electronic, Electronic Systems, and Electronics Systems. "IEEE Transactions on Power Electronics , 6 (1991), 281—286." 2002.

[73] Tang, Yonggang, Yuanxin Wu, Meiping Wu, Wenqi Wu, Xiaoping Hu, and Lincheng Shen. "INS/GPS integration: Global observability analysis". *IEEE Transactions on Vehicular Technology*, 58(3):1129–1142, 2009. ISSN 00189545.

[74] Temper, Dr. Dave. "DARPA Adaptable Navigation Systems (ANS)", 2010. URL https://www.darpa.mil/program/adaptable-navigation-systems.

[75] Titterton, D. and J. Weston. *Strapdown Inertial Navigation Technology, Second Edition*, volume 207. AIAA. ISBN 1563476932.

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

[77] Veth, Michael J. *Fusion of Imaging and Inertial Sensors for Navigation*. Ph.D. thesis, Air Force Institute of Technology, 2006.

[78] Win, Moe Z., Andrea Conti, Santiago Mazuelas, Yuan Shen, Wesley M. Gifford, Davide Dardari, and Marco Chiani. "Network localization and navigation via cooperation". *IEEE Communications Magazine*, 49(5):56–62, 2011. ISSN 01636804.

Vita

Major Gipson is a doctoral student at the Air Force Institute of Technology. He is a graduate of Rose-Hulman Institute of Technology, Webster University, the Air Force Institute of Technology and the Air Force Test Pilot School where he earned B.S.E.E., M.B.A., M.S.E.E. and M.S.F.T.E. degrees, respectively. His previous research includes air-to-air missile trajectory reconstruction and autonomous swarm control schemes for navigation support. His previous work includes various fighter aircraft flight test projects and leadership positions within several test squadrons.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704–0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202–4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
15—-2021	Doctoral Dissertation	Oct 2018–Sept 2021

4. TITLE AND SUBTITLE

An Enhanced Framework for Collaborative All-Source Navigation with Integrity

5a. CONTRACT NUMBER

in house

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Gipson, Jonathon S., Major, USAF

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB, OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENV-DS-21-S-065

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

13. SUPPLEMENTARY NOTES

This work is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

The Autonomous and Resilient Management of All-source Sensors with Stable Observability Monitoring (ARMAS-SOM) framework fuses collaborative all-source sensor information in a resilient manner with fault detection, exclusion, and integrity solutions recognizable to a Global Navigation Satellite System (GNSS) user. This framework uses a multi-filter residual monitoring approach for fault detection and exclusion which is augmented with an additional "observability" EKF sub-layer for resilience. We monitor the *a posteriori* 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 instances of ARMAS-SOM 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.

15. SUBJECT TERMS

cooperative, navigation, all-source, distributed

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Robert C. Leishman (ENG)
U	U	U	UU	211	19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636 robert.leishman@afit.edu