Improved Space Target Tracking Through Bias Estimation From In-situ Celestial Observations

A dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy at George Mason University

By

Thomas M. Clemons III
Master of Arts
US Naval War College, 1994
Electrical Engineer
Master of Science
US Naval Postgraduate School, 1989
Bachelor of Science
United States Naval Academy, 1982

Director: Dr. KC Chang, Professor
Department of Systems Engineering and Operational Research

Spring Semester 2010
George Mason University
Fairfax, VA

Copyright: 2010 by Thomas M. Clemons III
All Rights Reserved

DEDICATION

I dedicate this dissertation to my wife Martha and my two wonderful daughters Ellen and Sarah. You have been my inspiration and motivation throughout this process. Thank you for your understanding and patience.

ACKNOWLEDGEMENTS

I would like to thank the many friends, family and colleagues who have made this possible. I especially thank my loving wife Martha for believing in me even when I didn't believe in myself, and my daughters, Ellen and Sarah for their patience. My father and mother, whose encouragement and support gave me confidence to see it through. I acknowledge the members of my dissertation committee, Dr's Brian Mark, David Schum, and Alex Levis for their advice and wisdom as well as all the professors and teachers who contributed to my long academic career. Finally, thanks to Dr. KC Chang who provided invaluable advice, assistance, and encouragement throughout as my dissertation director and academic advisor.

TABLE OF CONTENTS

Page

LIST OF TABLES vii
LIST OF FIGURES viii
LIST OF ABREVIATIONS AND SYMBOLS x
ABSTRACT xii
1. INTRODUCTION 1
    1.1. Motivation 1
    1.2. Significance of the Problem 3
    1.3. Purpose of the Study 8
    1.4. Theoretical Framework 9
    1.5. Assumptions and Limitations 11
    1.6. Dissertation Organization 12
2. STATE OF THE ART 14
    2.1. Kinematic State Estimation 14
        2.1.1. The Kalman Filter 16
        2.1.2. The Linearized Kalman Filter (LKF) 17
        2.1.3. The Extended Kalman Filter (EKF) 17
        2.1.4. The Unscented Kalman Filter 18
    2.2. Sensor Fusion 19
    2.3. Other Linear Least Squares Estimation 19
    2.4. Bias Estimation and Correction 21
    2.5. Coordinate Frame Transformation 23
3. OVERALL APPROACH TO THE PROBLEM 27
    3.1. Bias correction algorithm development 27
    3.2. Simulation modeling 31
        3.2.1. Satellite Tool Kit® 32
        3.2.2. Target model 34
        3.2.3. Satellite and sensor model 34
4. BIAS MEASUREMENT AND CORRECTION 36
    4.1. Effect of sensor bias on target tracking 36
    4.2. Measuring sensor line of sight 39
    4.3. Determining background star observations 41
        4.3.1. Star catalog and star detection 41
        4.3.2. Rapid star identification algorithm 43
        4.3.3. Star vector rotation and field of view measurement 46
        4.3.4. Effect of aperture size 49

4.4. Bias modeling and measurement 50
    4.4.1. Bias model 50
    4.4.2. Measuring sensor bias 51
    4.4.3. Filtering the bias measurements 52
    4.4.4. Target observation correction 54
4.5. Bias error scenario development and validation 54
5. TARGET TRACKING AND ESTIMATION 61
5.1. Target location by vector intercept 61
5.2. Post correction target state estimation 63
    5.2.1. Space target dynamics 64
    5.2.2. Infrared sensor measurement model 65
    5.2.3. Linearized Kalman Filter 66
    5.2.4. Extended Kalman Filter (EKF) 69
    5.2.5. The Unscented Kalman Filter (UKF) 71
    5.2.6. Sensor Fusion 74
5.3. Post-correction target state estimator results 75
6. FILTER PERFORMANCE MEASURES 84
6.1. Posterior Cramer-Rao lower bound (PCLRB) performance 84
6.2. Filter Consistency 89
6.3. Test scenarios and validation process 91
7. SUMMARY AND FUTURE RESEARCH DIRECTIONS 95
7.1. Dissertation Summary 95
7.2. Implications of the results 97
7.3. Contributions 99
7.4. Study limitations 100
7.5. Recommendations for Future Research 101
APPENDIX A: SIMULATION RESULTS 104
APPENDIX B: SELECTED MATLAB PROGRAM FILES 154
APPENDIX C: ORBITOLOGY BASICS 176
REFERENCES 178

LIST OF TABLES

Table Page
Table 1. Mean error in target position for various bias and sensor noise values. 6
Table 2. Mean error in target position for various bias errors with constant sensor timing and position error. 6
Table 3. Research Variables 36
Table 4. Mean error in target position for various sensor noise values. 38
Table 5. Mean error in target position for constant sensor error 38
Table 6. Portion of nearest neighbor star catalog 45
Table 7. Mean error and number of stars with respect to aperture size 49
Table 8. Mean error in target position for pure sinusoidal bias 59
Table 9. Mean error in target position for constant bias with sinusoidal component 60
Table 10. Mean error in target position for pure sinusoidal bias 75
Table 11. Mean error in target position for constant bias with sinusoidal component 76
Table 12. Mean error in target position for 50 μrad sensor error (pure sinusoidal bias). . 79
Table 13. Comparison of mean error in target position for various system variables for common bias. 83
Table 14. Comparison of mean error in target position for non-filtered and filtered estimates. 83
Table 15. Target tracking scenarios 92
Table 16. Common test variables 93
Table 17. Average percent error improvement for all test scenarios 94

LIST OF FIGURES

Figure Page

Figure 1. The Space Tracking and Surveillance System as part of the BMDS architecture. 4

Figure 2. Mean error of target position for .02, .2, and 2 *m*rads bias error. 5

Figure 3. Mean error of target position using various nonlinear state estimators without bias correction as compared to non-corrected, non-filtered measurements. 7

Figure 4. Comparison of estimation filters in a nonlinear system. 18

Figure 5. Nominal target estimation and tracking architecture 28

Figure 6. Bias Correction and Tracking Algorithm Flow Chart 31

Figure 7. Screen shot of STK test scenario. 33

Figure 8. STK 3D display of tracking scenario. 33

Figure 9. Target and satellite trajectories 35

Figure 10. Mean error of target position for .2 *m*rads bias error in azimuth and elevation for each sensor. 39

Figure 11. Optical sensor measurement 40

Figure 12. Star map of 20 x 10 degree portion of the sky. 43

Figure 13. Star observation geometry 44

Figure 14. Star detection algorithm 46

Figure 15. Sensor trajectories with detected stars. 48

Figure 16. Mean error with respect to aperture size 49

Figure 17. Bias measurement in sensor field of view 52

Figure 18. Case 1 - Pure sinusoidal bias model 55

Figure 19. Case 2 - Constant bias with sinusoidal component model 56

Figure 20. Case 1 - Bias measurements and estimates for pure sinusoidal bias 57

Figure 21. Case 2 - Bias measurements and estimates for constant bias with sinusoidal component 57

Figure 22. Case 1 - Mean error of target position for sinusoidal bias (β_0 = 0.2 *m*rads) ... 59

Figure 23. Case 2 - Mean error of target position for a constant bias with a sinusoidal component (β_0 = 0.2 *m*rads) 60

Figure 24. Target position from the intercept of two vectors 62

Figure 25. Line of sight intercept geometry. 68

Figure 26. Range standard deviation over time 69

Figure 27. Two sensor fusion without feedback example. 74

Figure 28. Comparison of mean error of target position using LKF, EKF, and UKF estimation after EKF filter bias correction. 76

Figure 29. Mean error of target position with and without bias correction using LKF filter 78
Figure 30. Mean error of target position with and without bias correction using EKF filter 78
Figure 31. Mean error of target position with and without bias correction using UKF filter 79
Figure 32. Comparison of mean error of target position using LKF, EKF, and UKF estimation after EKF filter bias correction for 50 μrad sensor error........ 80
Figure 33. Comparison of mean error of target position using LKF, EKF, and UKF estimation after EKF filter bias correction for 100 μrad sensor error........ 80
Figure 34. Comparison of mean error of target position using LKF, EKF, and UKF estimation after EKF filter bias correction for 5 second measurement time step..... 82
Figure 35. Comparison of mean error of target position using LKF, EKF, and UKF estimation after EKF filter bias correction for 10 second measurement time step... 82
Figure 36. Error variance vs PCRLB for no bias case........ 88
Figure 37. Error variance vs PCRLB for .2 mrads bias error. 88
Figure 38. Normalized estimation error squared ($\boldsymbol{\epsilon}$) for LKF. 90
Figure 39. Normalized estimation error squared ($\boldsymbol{\epsilon}$) for EKF. 90
Figure 40. Normalized estimation error squared ($\boldsymbol{\epsilon}$) for UKF........ 91
Figure 41. Bias correction test program flow chart 92

LIST OF ABREVIATIONS AND SYMBOLS

ACS		Attitude Control System
BMDS		Ballistic Missile Defense System
DSP		Defense Support Program
ECF		Earth Centered Fixed Reference Frame
EKF		Extended Kalman Filter
FOV		Field of View
GPS		Global Positioning System
IAR		Inertial Attitude Reference
ICBM		Intercontinental Ballistic Missile
IMU		Inertial Measurement Unit
IR		Infrared
LKF		Linear Kalman Filter
LOS		Line of Sight
MDA		Missile Defense Agency
MHT		Multiple Hypothesis Tracker
MSE		Mean Square Error
NEA		Northeast Asia
NEES		Normalized Estimation Error Squared
OSM		Object Sighting Message
PCRLB		Posterior Cramer-Rao Lower Bound
rad		Radians
RV		Reentry Vehicle
RWA		Reaction Wheel Assembly
SBIRS		Space Based Infrared System
STD		Standard Deviation
STK		Satellite Tool Kit
STSS		Space Tracking and Surveillance System
TMA		Target Motion Analysis
UKF		Unscented Kalman Filter
x		State vector
X, Y, Z		Position components
A, B, G		System dynamics matrices
$f(x), F(x)$		System process function

u		Deterministic input
g		Gravitational constant (9.8m/s^2)
w		Process noise
P		State vector covariance matrix
z		Measurement vector
v		Measurement noise
Q		Process noise covariance matrix
H		Measurement matrix
K		Kalman gain
R		Observation error covariance matrix
β		Observation bias
ϖ		Orbit frequency
σ^2		Noise power spectral density (variance)
T_s		Time step
χ		UKF filter sigma vector
ζ		UKF filter sigma points
C		Rotation matrix
L, l		Target LOS vector
F		Focal length
θ		Azimuth
φ		Elevation
q		Quaternion
s		Star LOS vector
S		Sensor position
α		Line of sight difference angle
J		Jacobian matrix
ϵ		Normalized estimation error squared

ABSTRACT

IMPROVED SPACE TARGET TRACKING THROUGH BIAS ESTIMATION FROM IN-SITU CELESTIAL OBSERVATIONS

Thomas M. Clemons III, PhD

George Mason University, 2010

Dissertation Director: Dr. KC Chang

This dissertation provides a new methodology of using star observations and advanced nonlinear estimation algorithms to improve the ability of a space based Infrared tracking system to track cold body targets in space. Typically, the tracking system consists of two satellites flying in a lead-follower formation tracking a ballistic or space target. Each satellite is equipped with a narrow-view IR sensor that provides azimuth and elevation measurements to the target. The tracking problem is made more difficult due to a constant, non-varying or slowly varying bias error present in each sensor's line of sight measurements. The conventional sensor calibration process occurs prior to the start of the tracking process and does not account for subsequent changes in the sensor bias. This dissertation develops a technique to estimate the sensor bias from celestial observations while simultaneously tracking the target. As stars are detected during the target tracking process the instantaneous sensor pointing error can be calculated as the difference between a measurement of the celestial observation and the known position of the star.

The system then utilizes a separate bias filter to estimate the bias value based on these measurements and correct the target line of sight measurements. The study develops and compares the ability of three advanced nonlinear state estimators: A Linearized Kalman Filter; an Extended Kalman Filter; and an Unscented Kalman Filter, to update the state vector. The bias correction-state estimation algorithm is validated using a number of scenarios that were created using The Satellite Toolkit©. The variance of the target position error resulting from the nonlinear estimation filters is compared to the posterior Cramer-Rao lower bound and a filter consistency check. The results of this research provide a potential solution to sensor calibration while simultaneously tracking a space borne target with a space based sensor system.

1. INTRODUCTION

In this dissertation we provide a new methodology of using star observations and advanced nonlinear estimation algorithms to improve the ability of a space based Infrared tracking system to track cold body targets in space. The methodology uses the stars seen in the sensor field of view to estimate the sensor's bias and calibrate the sensor while simultaneously tracking the target. Once bias error is removed from the target measurement an advanced nonlinear estimation filter improves the target state vector estimate.

1.1. Motivation

Missile defense and space situational awareness are becoming critical in today's military operations and improving these capabilities is an important area of study. One limitation of today's systems in these missions is that land based radar systems are not always in a position to continuously track space and ballistic missile targets. Due to geographic limitations space objects and ballistic missiles are not continuously visible during long portions of their trajectories. This shortfall decreases the effectiveness of missile defense and space situational awareness. To help overcome this challenge aggressive design and testing is underway on space-based sensors that will provide tracking of space objects and ballistic missiles where the ground radars are not able to provide.

Using higher altitude to gain advantage in surveillance is a fundamental part of military theory. Prior to the advent of flight militaries attempted to gain this advantage by placing observation posts on a hill or mountain side. In the 1800's militaries began using balloons to observe enemy activities. One of the primary purposes of the airplane during WWI was to act as an observation platform. Evolution of flight led to higher and faster aircraft until, during the latter half of the 20th century, space became the ultimate high ground with spy satellites providing access to denied areas for reconnaissance and early warning. At the dawning of this century there is an effort to use the advantage of this space "high-ground" to provide tracking of other space-borne objects such as ballistic missiles and other satellites.

A space-based tracking system provides many advantages for missile defense and space situational awareness. As a part of a greater ballistic missile defense system of systems that contribute to an overall picture, an in-situ system utilizing an IR sensor will cover gaps in terrestrial radar coverage, allow interceptors to engage enemy missiles earlier in their trajectories, discriminate between warheads and decoys, and provide warhead hit assessment. However, systemic errors in sensing systems hinder accurate threat identification and target location, and, in this way, the space-based tracking systems present some unique challenges.

An example of such a system is the Missile Defense Agency's Space Tracking and Surveillance System (STSS). The STSS test bed, recently launched in September 2009, consists of two spacecraft in a lead/trail configuration in the same low earth orbit (1000 – 1500 km). The STSS will test the ability of tracking threat missiles against the cold background of space, one of the biggest challenges of ballistic missile defense. To accomplish this mission, each spacecraft will consist of three main components: a wide-view acquisition sensor, a narrow-view tracking sensor, and a signal and data processor subsystem.

In a ballistic missile tracking scenario, the wide-view acquisition sensor provides high-resolution horizon-to-horizon surveillance capability to detect an enemy ballistic missile just after launch - when its rocket engines are burning hot. The STSS may also receive cues from other sensors in the missile defense architecture. Once the enemy missile has completed its boost phase and passed into its midcourse phase, the narrow-view tracking sensor will pick up the threat and follow it through the cold vacuum of space. Even though a midcourse phase ballistic missile will not have heat producing rocket plumes, the cryogenically cooled narrow-view tracking sensor will detect the thermally dim warhead [1]. In a space situational awareness scenario, the STSS will use externally provided ephemeris data to direct the narrow-view sensor for target detection. Figure 1 shows how the STSS will fit into the overall Ballistic Missile Defense System (BMDS) architecture. (See Appendix C for a description of basic orbits and the phases of ballistic missile flight).

1.2. Significance of the Problem

In addition to the tasks listed in Figure 1, to accurately track and report targets the sensing spacecrafts and BMDS must also perform the following functions:

- Transfer tracks from the wide-view acquisition sensor to the narrow-view tracking sensor.

- Associate tracks from other elements of the BMDS including Space Based Infrared Sensor (SBIRS) system and Defense Support Program (DSP) spacecrafts with its own tracks.
- Estimate and remove bias errors inherent in the spacecraft and its sensor (sensor calibration).
- Provide a quantitative measure of the track quality through covariance data.

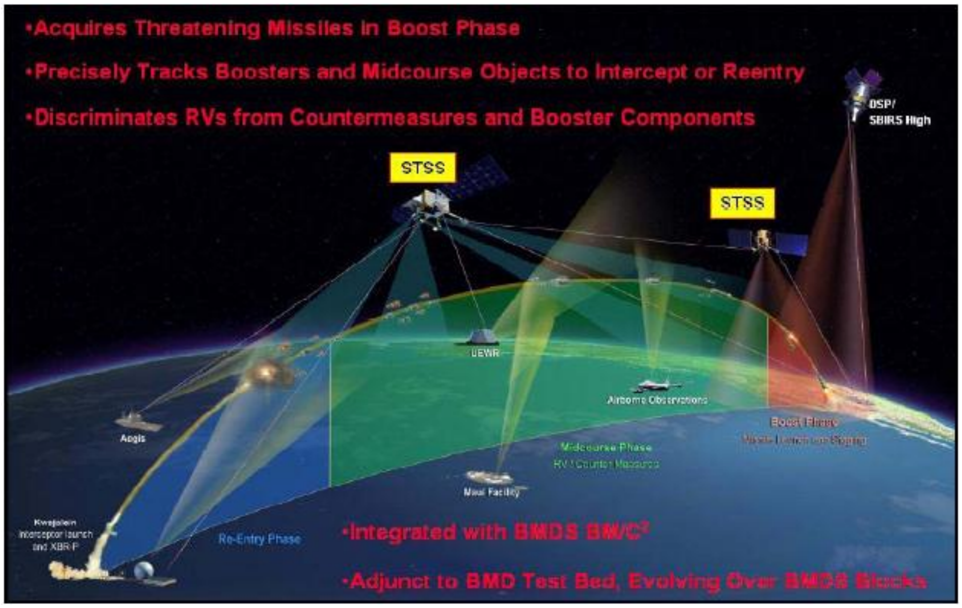

•Acquires Threatening Missiles in Boost Phase
•Precisely Tracks Boosters and Midcourse Objects to Intercept or Reentry
•Discriminates RVs from Countermeasures and Booster Components
DSP/
SBIRS High
STSS
STSS
UEWR
Airborne Observations
Boost Phase
Aegis
Midcourse Phase
RV / Counter Measures
Naval Facility
Kwajalein
Interceptor launch
and XBR-P
Re-Entry Phase
•Integrated with BMDS BM/C²
•Adjunct to BMD Test Bed, Evolving Over BMDS Blocks

Figure 1. The Space Tracking and Surveillance System as part of the BMDS architecture [2].

Of the items listed above, sensor calibration is a common requirement to adequately perform the others and is an area of limited research. Bias error in a spacecraft and sensors can result from a number of different sources, including:

- Errors in spacecraft position (spacecraft navigation bias)

- Errors in spacecraft attitude (wheel assembly controller error, coordinate system translation round-off error)
- Errors in sensor calibration (residual pointing error, degradation of sensor alignment)
- Errors in timing caused by bias in the clocks of the sensors

An analysis of sensor pointing and timing bias contributions to errors in target location is shown in Figure 2 and Table 1 for a nominal scenario. Here we see the magnitude of the mean error from the true target position caused by bias error. The plot shows the mean error over time for four constant bias values in the sensor azimuth reading of the lead satellite in a typical tracking scenario. In general, as bias increases, the position error of the target increases proportionally to the range from the target.

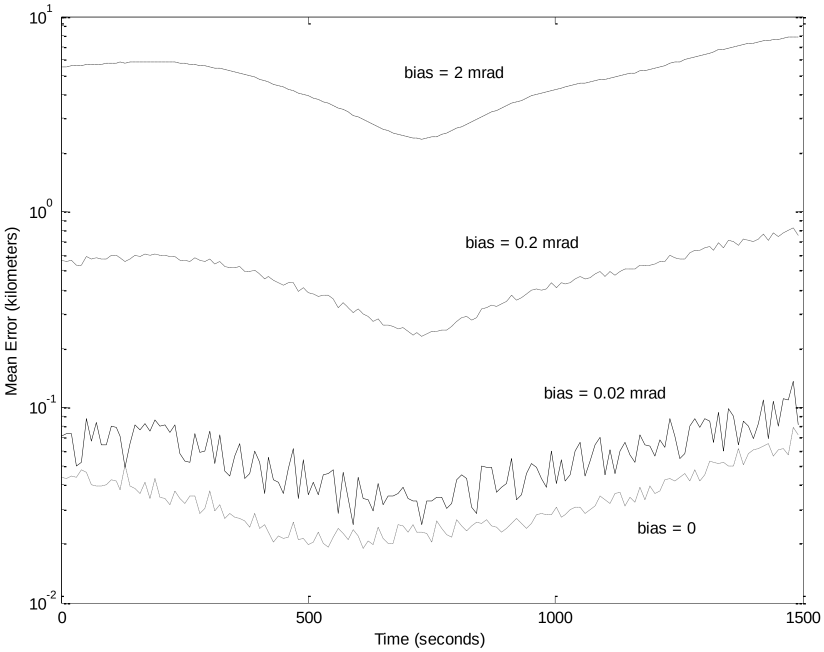


Figure 2. Mean error of target position for .02, .2, and 2 mrads bias error.
(Sensor error = 10 μrad, Process error = 10 meters)

For the nominal scenario Table 1 provides a comparison of error measurements for various values of bias and filter sensor error. As sensor error increases it becomes the dominating factor.

Table 1. Mean error in target position for various bias and sensor noise values.

Bias	$\sigma_s = 5\ \mu$rad	$\sigma_s = 10\ \mu$rad	$\sigma_s = 20\ \mu$rad	$\sigma_s = 1m$rad
None	21.6 m	33.8 m	62.2 m	3.01 km
.02 *m*rad	42.5 m	49.6 m	72.6 m	3.01 km
.2 *m*rad	373.6 m	374.5 m	377.5 m	3.02 km
2 *m*rad	3.72 km	3.72 km	3.72 km	4.72 km

To explore the effect of timing error, a typical delay of 10 *m*s in the first satellites measurements was introduced. Table 2 shows the effect of this error compared with the error from Table 1. It is seen here that timing bias error increases the position error, but to a lesser extent than bias in the observation. One can see that for a given bias error, the timing bias has less effect than an increased measurement angle bias.

Table 2. Mean error in target position for various bias errors with constant sensor timing and position error.

Bias	$\sigma_s = 10\ \mu$rad	$\Delta t = 10\ ms$
None	33.8 m	61.6 m
.02 *m*rad	49.6 m	66.1 m
.2 *m*rad	374.5 m	383.7 m
2 *m*rad	3.72 km	3.73 km

These numbers show that beyond the most minor sensor bias error tested (.02 *m*rad) the significance of the error in target position is of great concern to the ability of the system to reliably locate the target. The ability of the missile defense system to accurately map the target cluster for the warhead, or to associate tracks, would be hampered by this magnitude of error. Additionally, in a space situational awareness scenario, a 375 meter error in the case of .2 *m*rad sensor bias may result in a miss-leading conjunction analysis that could lead to a satellite collision.

Figure 3 shows the results from an attempt to use nonlinear estimation to improve the target state estimate without correcting for bias error. The black line represents the target position error without bias correction or nonlinear estimation. The colored lines represent the error after filtering the uncorrected observations. As one can see there is some improvement in target tracking accuracy, but significant error still exists. For instance, the EKF and UKF begin to improve the target position around the 600 and 1200 second points, but eventually degrade back toward the uncorrected values. The LKF provides only marginal improvement in reducing the magnitude of the error variance.

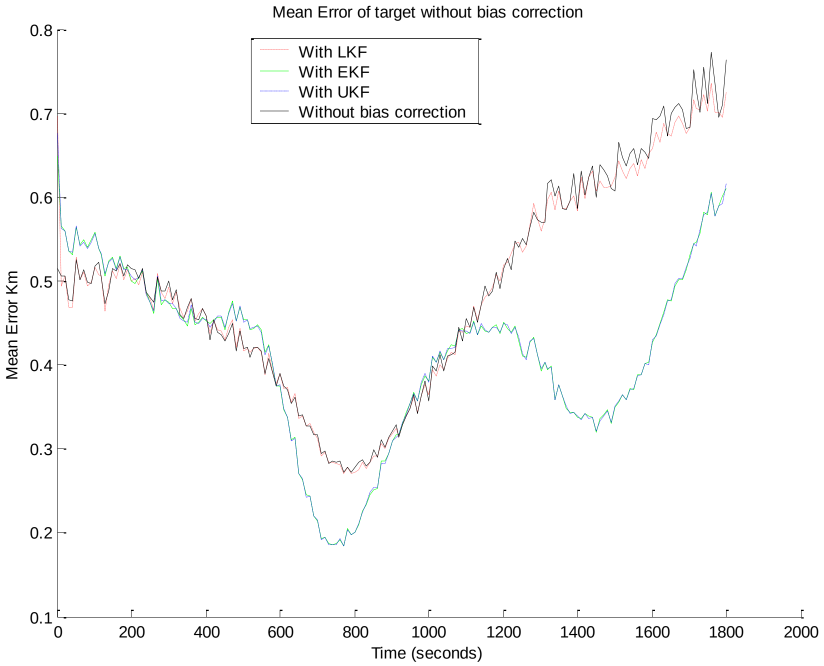


Figure 3. Mean error of target position using various nonlinear state estimators without bias correction as compared to non-corrected, non-filtered measurements.

1.3. Purpose of the Study

The current method of calibrating the narrow view sensor occurs prior to target tracking. Sometime prior to starting the tracking process the narrow view sensor is directed to focus on an identified star. The system then attempts to measure and correct the bias present at that time. Once this calibration is complete, the sensor begins tracking the target with no further updates to the bias measurements. Some systems perform this calibration on a periodic basis. These methods are relatively short-lived and do not account for continued changes to the sensor bias due to effects such as heating and cooling of the spacecraft.

The purpose of this dissertation is to assist in the improvement of space-based tracking of space objects, particularly ballistic missiles in un-powered mid-course phase of flight and unidentified satellites and space debris that could lead to collisions with other active satellites. To achieve our goal, we develop a methodology that will simultaneously track a target and correct for bias errors. Specifically, this research was conducted to answer the following questions:

1. How do error sources affect target tracking?
2. Can bias error be removed through background star observations?
3. Which state of the art methodology will provide best estimation of target state and accommodate sensor bias error?
4. What is the performance bound of the algorithm under the influence of various degrees of biases?

1.4. Theoretical Framework

A single-target tracking IR system will typically attempt to keep the target centered of the sensor field of view and provide measurements of target line of sight (LOS) angles to an algorithm that estimates a target state such as position, velocity, and acceleration. Limitations on using infrared sensors for tracking include: 1) false target detection requires resolution of ‘ghost’ tracks; 2) very accurate sensor pointing is required for single-target tracking and sensor and platform bias error greatly affect target state determination; and 3) bearing only measurements make it difficult to determine target state and changes to the state vector. Methods to enhance sensor LOS accuracy and resolution of target state dynamics will minimize track uncertainties and enhance track state estimation. Systemic errors in sensing systems hinder accurate threat identification and target location, in this way the space based tracking systems present some unique challenges.

Typically two sources of error exist in a target tracking problem. The first is error due to random processes in the space tracking system and are commonly called system dynamic and measurement error. These errors are typically Gaussian with zero mean and certain variance. The second source is a systemic error that causes all measured values to deviate from the true value by a consistent or slowly varying amount in a constant or near-constant direction. This error is what we refer to as the bias error.

Both these errors, if uncorrected or insufficiently characterized, can lead to significant deviations in the perceived location of the target object. Bias error in a spacecraft sensor can result from a number of different sources include those discussed in the previous section. In a missile defense scenario, an interceptor must know the location of the target with sufficient accuracy so the onboard terminal homing sensor can locate the target within its field of view. When one considers that the target and sensor are both moving at orbital speeds this becomes a stressing problem.

Using a space-based IR system to track another space object involves the following areas of concern:

1. *Accurate spacecraft position determination:* The space tracking mission requires real-time knowledge of spacecraft position and the baseline between two or more spacecraft for precise formation of target location.
2. *Accurate spacecraft attitude determination:* The attitude control system (ACS) of the spacecraft must know the present attitude of the spacecraft to allow transformation of the sensor azimuth and bearing measurements into a common coordinate frame for calculation of the target position.
3. *Sensor pointing accuracy:* The sensor pointing control requirements are driven by the need to ensure that a target warhead falls within the field of view of the narrow-view sensor and that the sensor provides true, unbiased, error-limited bearings to the target. The sensor pointing accuracy is a function of the spacecraft pointing accuracy described above, the calibration and alignment of the sensor, and changes caused by spacecraft dynamics such as thermal distortion, flutter and vibration.
4. *Modeling of spacecraft dynamics:* Effects on the spacecraft from various sources such as thermal distortions in the bus, slew induced structural vibration, reaction wheel assembly (RWA) controller error, RWA-induced vibration, thermal flutter,

and others dependent upon spacecraft design. In general these effects can be considered sources of error in the pointing accuracy of the payload sensors. Efforts are made in design of the spacecraft to reduce these effects to very small values. However, residual effects might not be insignificant. These errors are assumed to be slowly varying with a zero mean and part of the residual errors they affect.

5. *Coordinate frame translation:* There are a number of coordinate frames involved in tracking targets with a space-borne sensor system [3]. A typical observation by the spacecraft sensors will require a number of transformations through these coordinate systems to arrive at a common position in the inertial reference frame. Transformation between these reference frames result in nonlinearities and round-off errors.

1.5. Assumptions and Limitations

This study is limited to characterizing and correcting errors associated with targets that are already in track. It does not attempt to deal with the problem of target detection, track initiation, track identification, or transition from detection to tracking. We assume that the target is already in track and that it was identified as a target of interest. Further, although the system is part of a greater missile defense and space situational awareness system, in this study the system does not include tracking data from other sensors such as terrestrial radars. Finally, although the results could represent any bearing only tracking system, the algorithms and techniques in this study are optimized for a spacecraft mounted sensor tracking ballistic missiles or other space objects.

As mentioned above, errors in satellite attitude will contribute to target location error in much the same way as bias error. This occurs through the coordinate transformations required to transfer observations from the sensor frame into the inertial reference frame. However, attitude control error detection and correction has been studied extensively and control systems using gyro rate measurements along with star sensors provide outstanding correction to attitude errors such that their contribution to this problem is assumed to be negligible.

1.6. Dissertation Organization

This dissertation presents the results of the study as follows. Chapter 2 discusses the current state of the art and literature relevant to our research. The chapter consists of five sections. The first section provides a general overview of the target tracking and estimation including a description of the Kalman Filter and some nonlinear techniques. The second is an overview of sensor fusion. The third section reviews some bearing only tracking methodologies applicable to the space-based tracking problem. The fourth section focuses on bias estimation and correction. The chapter is concluded with a description of the coordinate frames used in the study.

Chapter 3 is an outline of our approach to solving the problem of improving space based target tracking. We begin with a broad overview of the six steps in our approach to developing the bias correction and track improvement algorithms. The second section provides an overview of the simulation models used to test the algorithms.

Chapter 4 provides the bias measurement and correction algorithm development and testing. First is an explanation of the impact that bias plays in space based target tracking with results from the first phase of the study. The second section details the function of the IR sensor. Next we develop the methodology used to determine if a star observation is expected and how the error is measured using the stars. In section 4 a bias model is constructed and a comparison of solutions to finding the bias estimate from the bias measurements is undertaken using a simple sample and hold method and an Extended Kalman Filter. The bias estimate is then used to correct the observations.

Chapter 5 focuses on using these corrected observations to obtain a refined target estimate through three methods of nonlinear target state estimation; a Linearized Kalman Fitler (LKF), an Extended Kalman Filter (EKF), and an Unscented Kalman Filter (UKF). The theory behind each of these filters is developed and tested with a comparison scenario.

Chapter 6 focuses on performance analysis of the tracking algorithm. It reviews two theoretical performance measures and provides an analysis of our system in performing improved target tracking. The second part of the chapter discusses the experimental design as well as our results. Twelve different scenarios involving missile and satellite tracking validate the developed methodology. Results for these scenarios are summarized and compared.

Chapter 7 concludes the study with a summary of the key insights learned as well as future work.

2. STATE OF THE ART

From the early sixties there has been increasingly steady progress in the tracking of orbital spacecraft and ballistic objects using both radar and optical systems. Most previous study involved the use of terrestrial systems for civil and military applications. The civil space program required a world-wide network of tracking stations to maintain inventory of satellites and manned spacecraft. However, the Cold War with the Soviet Union and the treat of Inter-Continental Ballistic Missile (ICBM) attack lead both countries to develop anti-ballistic missile defenses and warning networks. The efforts made in both these areas contributed to significant advances in tracking capability. Most of this work was in tracking of ballistic missile re-entry vehicle (RV), or warhead, and re-entry of satellites and debris. RV tracking involves tracking a target as it departs the exo-atmospheric phase of flight and enters the drag-inducing atmosphere where deceleration from aerodynamic effects takes over [4]. In this study, our concern is with tracking while the target is still in the exo-atmosphere where the atmospheric effects are small and the target is difficult to track.

2.1. Kinematic State Estimation

The objective of target tracking is the collection of "sensor data from a field of view containing one or more potential targets of interest and then partition the sensor data into sets of observations, or tracks that are produced by the same sources." [5] Modern sensor systems include electro-magnetic (such as radar and radio direction finding), electro-optical (visual and infrared (IR)), and acoustic (such as sonar). These systems can be either single-target trackers, or multiple-target trackers as required. A single-target tracking IR system will typically attempt to keep the target centered of the sensor field of view and provide measurements of the line of sight (LOS) angles to an algorithm that attempts to estimate the target state such as position, velocity, and acceleration [5]. Limitations on using infrared sensors for tracking include: 1) false target detection requires resolution of ‘ghost’ tracks, 2) very accurate sensor pointing is required for single-target tracking sensors [5] and sensor and platform bias error greatly affect target state determination, and, 3) bearing only measurements make it difficult to determine target state and changes to the state vector. Methods to enhance sensor LOS accuracy and resolution of target state dynamics will minimize track uncertainties and enhance track state estimation.

State estimation of a missile target has been the subject of intense study for over four decades. Tracking filters of various types have been used for this purpose since the early days of the discovery of the Kalman Filter [6]. A good overview of tracking techniques as they apply to ballistic missile targets was presented by Farina et al in [7]. They specifically used the Kalman Filter, Extended Kalman Filter, and Particle Filter to the problem of a ballistic object during reentry phase of flight using radar measurements. In some cases, the problem is the subject of large sections of numerous texts including [5] and [8, 9].

2.1.1. The Kalman Filter

The Kalman filter is the optimal linear filter for recursive estimation of a state in the presence of Gaussian noise [10]. The discrete Kalman filter begins with a model of the system and measurement processes. In its generalized form, the system model is described by:

$$x_{k+1} = Ax_k + Bu_k + Gw_k \tag{2-1}$$

where x_k is the state vector at time k, A is the state transition matrix, u is an external force acting on the system, and w is the white process noise with covariance Q. The measurements of the system are modeled by:

$$z_k = Hx_k + v_k. \tag{2-2}$$

where v_k is the white measurement noise with covariance R.

Given a set of initial state conditions x_0, and an initial state covariance matrix P_0, a prediction of the next state and its covariance is given by

$$\hat{x}_{k|k-1} = A\hat{x}_{k-1|k-1} \tag{2-3}$$

$$P_{k|k-1} = AP_{k-1|k-1}A^T + Q. \tag{2-4}$$

The state estimate is updated by

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k\left(z_{k|k-1} - H_k\hat{x}_{k|k-1}\right) \tag{2-5}$$

$$P_{k|k} = P_{k|k-1} - K_kHP_{k|k-1} \tag{2-6}$$

Where K is the Kalman gain matrix

$$K_k = P_{k|k-1}H^T\left[HP_{k|k-1}H^T + R\right]^{-1}. \tag{2-7}$$

However, because the space tracking problem is nonlinear both the state and measurement models, the Kalman Filter is not the optimal filter for this problem.

2.1.2. The Linearized Kalman Filter (LKF)

The conversion of object measurements from one coordinate system to another requires the conversion of the measurement equations from a nonlinear format to a linear format, much the same way one transforms spherical coordinates into Cartesian coordinates. This conversion effort essentially linearizes the measurement error covariance matrix and requires that the matrix be conditioned by the coordinate transformation. This methodology is further explored in section 3.7.1

2.1.3. The Extended Kalman Filter (EKF)

The EKF is widely used in nonlinear vector-observation such as the system explored here. The EKF approach applies the standard Kalman Filter (for linear systems) to nonlinear systems with additive white noise by continually updating a linearization around the previous state estimate, starting with a given initial estimate. In this way the nonlinear equation is linearized by ignoring or approximating high-order terms. This is accomplished through partial derivatives of the state transition matrix $f(x)$, and the measurement matrix $h(x)$ with respect to the state x, in an attempt to linearize the nonlinear dynamic observation equations. The derivative is performed at each time step, k, resulting in the Jacobian matrix, designated $F(x_k)$ and $H(x_k)$. Once the partial derivatives are found, standard Kalman Filtering procedures are used to determine the predicted state and its covariance.

2.1.4. The Unscented Kalman Filter

The concept behind the UKF algorithm is the estimation of the mean and covariance through propagation of the state and covariance matrix and a state update with observation information, similar to the standard Kalman filter [11]. However, in the UKF, a set of sigma points, approximating the distribution of the state, are propagated through the state transition and observation functions and averaged to find a new mean and covariance. In this way, the UKF avoids the requirement to calculate the Jacobians and achieve second-order approximation of the nonlinear equations. Figure 4 shows a comparison of EKF and UKF estimation filters.

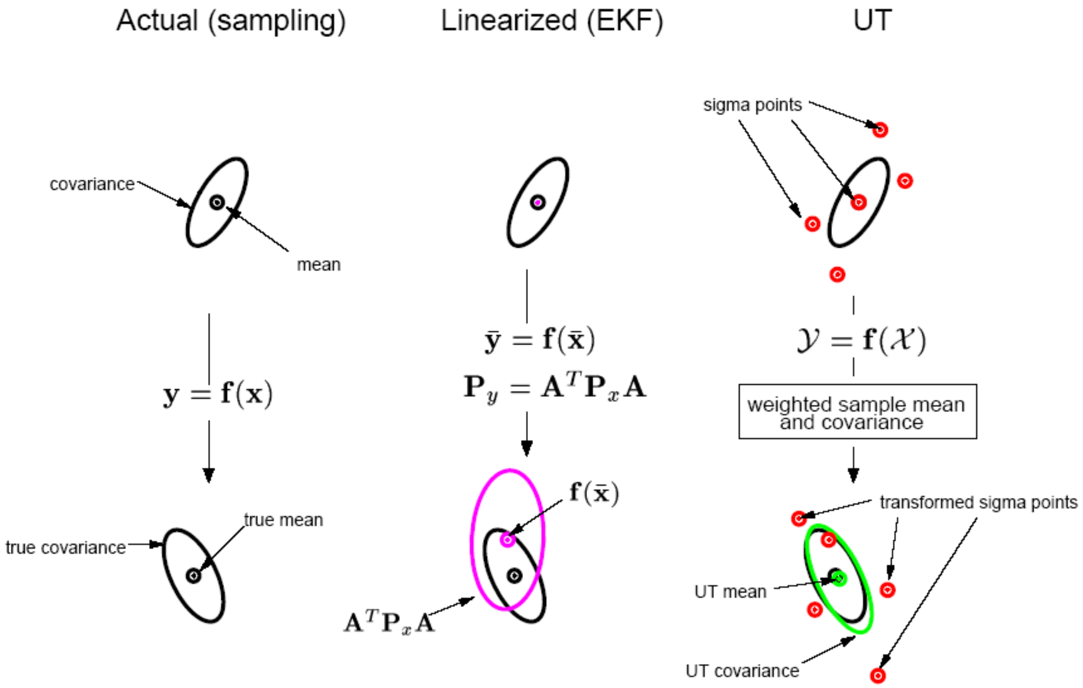

Actual (sampling)
Linearized (EKF)
UT
covariance
mean
$\mathbf{y} = \mathbf{f}(\mathbf{x})$
true mean
true covariance
$\bar{\mathbf{y}} = \mathbf{f}(\bar{\mathbf{x}})$
$\mathbf{P}_y = \mathbf{A}^T\mathbf{P}_x\mathbf{A}$
$\mathbf{f}(\bar{\mathbf{x}})$
$\mathbf{A}^T\mathbf{P}_x\mathbf{A}$
sigma points
$\mathcal{Y} = \mathbf{f}(\mathcal{X})$
weighted sample mean and covariance
transformed sigma points
UT mean
UT covariance

Figure 4. Comparison of estimation filters in a nonlinear system.

2.2. Sensor Fusion

Another study exploring nonlinear estimation for ballistic missile tracking looked at fusing tracks from multiple sensors [12]. This fusion technique is applicable to the current problem since there may be two or more satellites in the tracking architecture, which would be centralized or distributed. Two basic forms of sensor fusion are fusion with and without feedback. In our system we assume that the sensor fusion is occurring far from the sensor satellites and that feedback will not occur. Additionally, sensors such as radars could be included into the fusion architecture as they are available to provide an even more refined picture.

2.3. Other Linear Least Squares Estimation

There are a number of studies that used iterative versions of a linear least square estimation for target tracking. One such technique for angle only measurements of ballistic missiles in exo-atmospheric flight is presented in [13]. In that study Chang determined that the least square filter better achieves the Cramer-Rao bound than the EKF. The iterative algorithm processes a batch (N points) of data in an attempt to minimize the weighted least square error. In this way it is different from the recursive EKF. This technique is computationally difficult to apply due to the iterative nature of the problem and the sensitivity of the initial guess.

Murali Yeddanapudi et al in [14] also studied state estimation of a target in exo-atmospheric flight using space-based line of sight measurements. Their effort uses a more sophisticated Levenberg-Marquardt iterative algorithm to estimate the initial state from LOS observations in place of a simpler Gauss-Newton algorithm, which is limited by accuracy of an initial guess of the target state. This algorithm uses a six-dimensional state vector and is limited in that it assumes there are no accelerations other than gravity acting on the target.

Limitations of the Extended Kalman Filter in Target Motion Analysis (TMA) for general bearings-only tracking applications are explored by Ailada in [15]. His research determined that interaction between bearing inaccuracy can cause covariance matrix collapse and subsequent filter divergence. He suggested that substantial improvements in filtering stability can be realized by employing alternative initiation and relinearization procedures. He suggests several methods to decouple the covariance computations from the estimated state vector to prevent feedback and amplification of errors. The effect of his proposal forms a pseudolinear TMA algorithm.

Alfonso Farina in [16] and Rao in [17, 18] present some fundamental target tracking algorithms required for estimation of target position and velocity for bearings only tracking. The use of Maximum Likelihood estimation is used in [19] to find target acceleration and position and velocity estimation for bearings only observations from a single observer are found in [20] where a linear least squares estimator with a closed-form solution was proposed but results in severe bias problems. While the techniques used in [15-20] are related to this study, they all deal with tracking non-maneuvering targets in two dimensions in an undersea environment using Target Motion Analysis techniques. The observer must make a number of significant maneuvers to change aspect and target bearing rate to achieve stable results. This is not the case with space systems and limits their applicability to our problem; however, they do provide some useful insight into the mathematical models involved.

2.4. Bias Estimation and Correction

There are several methods of bias correction in the literature that apply to tracking systems. A particular problem involves estimating the state of a linear dynamic system influenced by a constant, but unknown bias vector. As long as the bias is measurable in some respect, a method of handling this problem is through state augmentation, where the bias vector is appended to the state vector. This method, however, may result in computations involving large matrices if the number of bias terms is greater than the number of state terms with the inaccuracies associated with computations of large matrices. Another technique to estimate a state affected by a bias vector separates the estimation of the bias from the estimation of the dynamic state in a two-stage estimation filter as described in [21]. Therein, the author, Bernard Friedland, provides a method of determining the state of a linear process in the presence of a constant, but unknown bias vector. In this technique, two separate, uncoupled, Kalman Filters run in parallel to generate the estimate of the "bias-free state" and the bias estimate. The estimated state is then corrected using the simultaneously calculated bias estimate. This bias estimation concept is further explored by Friedland and others in [22-25]. The first paper modifies the process to allow infrequent observations of the bias. The second and third papers consider the case of a slowly varying, Gaussian, white noise random bias. The final paper, Friedland looks at bias correction when there are noise-free measurements. These bias correction techniques appear to be applicable for systems where the bias operates on the state vector in some additive manor, such as attitude control or inertial navigation systems. Although these systems are present in the satellite tracking problem we are considering, we are more concerned with correction of bias in the observations.

The correction of bias in space-based tracking utilizing two satellites in formation is similar to the sensor registration problem associated with more traditional terrestrial based radar multi-sensor integration problem. There has been a significant amount of work in this area and a few recent studies provide some related concepts [see 7, 26-30]. Of these, two techniques for bias modeling and estimation for airborne Ground Moving Target Indicator radar sensors are given in [29] and [30]. In the first, the authors incorporate a bias filter that uses an EKF bias estimation algorithm that takes the measurements from the Multiple Hypothesis Tracker (MHT) detections of 'static' reference point to estimate and correct the biases associated with platform location, heading, and velocity errors. The second paper uses non-moving static targets of unknown locations to correct for sensor platform location errors. Bar Shalom in [27] develops an algorithm for sensor registration of two 3-D sensors prior to track integration. Although the paper provides worthwhile mathematical development of sensor/target registration geometry, its application is limited to the special case of closely located sensors (on the same platform) with no position error. In addition, the sensors must be able to correlate the target being used for bias estimation, which may not be the case in a multi-target environment.

2.5. Coordinate Frame Transformation

There are several other studies that involve correction of error bias that are unrelated to missile tracking, yet provide beneficial insight into the problem. Foremost among these is the paper by Kistosturian in [31] where he describes in detail the errors associated with a spacecraft antenna pointing system. He also discusses the many coordinate frame transformations required in the tracking of space based objects. The goal is to transform the measured azimuth and elevation of the target as read by the tracking sensor into a common reference frame that can be used by the system to locate the target and fuse that location estimate with other systems. The errors in the spacecraft attitude and sensor pointing angles are transformed along with the observation as a function of an incremental rotation matrix. Spatial orientations of coordinate frames associated with a satellite tracker and its orbit are defined by their respective coordinate transform matrices as explained in [31] for a communications antenna. We modify them here for the case of an optical sensor. Their definitions and transformation matrices are:

a. *Earth Center Inertial (ECI) Coordinate Frame (I-Frame)*: A non-rotating, earth-based coordinate frame fixed at the center of the earth with its X-axis defined by a celestial point in space. This system provides the frame for celestial observations and is used as the baseline coordinate frame that all others are referenced to. Its $\boldsymbol{U}_{X,I}/\boldsymbol{U}_{Y,I}/\boldsymbol{U}_{Z.,I}$ unit vectors point as follows:

   $\boldsymbol{U}_{X,I}$: in the direction of the vernal equinox or first point of Aries.

   $\boldsymbol{U}_{Z.,I}$: in the direction of the Earth's North Pole

   $\boldsymbol{U}_{Y,I} = \boldsymbol{U}_{Z.,I} \times \boldsymbol{U}_{X,I}$

b. *Orbital Coordination Frame (O-Frame)*: Also known as the Peri-focal coordinate frame, the orbital coordinate frame is centered on the satellite center of gravity and has the corresponding unit vectors defined as follows:

$\boldsymbol{U}_{ZO,I}$: to the center of the Earth

$\boldsymbol{U}_{YO.,I}$: opposite to the direction of the satellite angular momentum

$\boldsymbol{U}_{XO,I} = \boldsymbol{U}_{YO.,I} \times \boldsymbol{U}_{ZO,I}$ (tangential to the orbit in direction of the satellite velocity)

The orbits associated with space tracking systems are explained in Appendix C.

Orbital Elements: The location of the satellite in the orbit is determined by six classical orbital elements defined as follows:

1. a, *semi-major axis* - a constant defining the size of the orbit
2. e, *eccentricity* – a constant defining the shape of the orbit. A circular orbit has $e=0$.
3. i, *inclination* – the angle between the equatorial plane and the orbital plane.
4. Ω, *longitude of the ascending node* – the angle, in the equatorial plane between the X-axis and the point where the satellite crosses through the equatorial plane in a northerly direction.
5. ω, *argument of periapsis* – the angle, in the orbital plane, between the ascending node and the periapsis point, measured in the direction of the satellites motion.
6. T_p, *time of periapsis passage* – the time when the satellite was at periapsis.

From these elements the transformation matrix from the O-Frame to the I-Frame is computed [32].

$$C_{Orb}^{ECI} = \begin{bmatrix} R_{11} & R_{12} & R_{13} \\ R_{21} & R_{22} & R_{23} \\ R_{31} & R_{32} & R_{33} \end{bmatrix} \tag{2-8}$$

where

$$\begin{aligned}
R_{11} &= \cos\Omega\cos\omega - \sin\Omega\sin\omega\cos i \\
R_{12} &= -\cos\Omega\sin\omega - \sin\Omega\cos\omega\cos i \\
R_{13} &= \sin\Omega\sin i \\
R_{21} &= \sin\Omega\cos\omega + \cos\Omega\sin\omega\cos i \\
R_{22} &= -\sin\Omega\sin\omega + \cos\Omega\cos\omega\cos i \\
R_{23} &= -\cos\Omega\sin i \\
R_{31} &= \sin\omega\sin i \\
R_{32} &= \cos\omega\sin i \\
R_{33} &= \cos i
\end{aligned}$$

c. *Satellite Body Coordinate Frame (B-Frame)*: This coordinate frame has the same origin as the O-Frame, but is fixed to the body of the satellite. It rotates about the origin in accordance with the satellite attitude. Typically the attitude of the B-Frame referenced to the O-Frame is defined by roll, pitch, and yaw and is controlled by the satellite's attitude control system. The rates of rotation are measured by the rate gyros fixed to the satellite body.

d. *Sensor Coordinate Frame (S-Frame)*. Equivalent to as the A-Frame in [31], the S-Frame is the sensor coordinate frame. It is the reference frame to measure the sensor pointing angles, elevation and azimuth. Generally it is fixed to the B-Frame and its transformation matrix is computed from its location and orientation of the sensor support structure on the satellite body.

e. *Telescope Pointing Frame (T-frame).* The T-Frame represents the line of sight (LOS) of the narrow view sensor. The telescope Z_T unit vector, $\boldsymbol{U}_{T,Z}$, points in the same direction as the sensor boresight axis. When azimuth and elevation are zero it aligns with the S-Frame. Target locations within the field of view of the sensor are computed based on this coordinate frame.

f. *The Earth-Centered, Earth-Fixed Coordinate Frame (ECEF).* A geocentric system that rotates with the earth. This system is usually used to represent terrestrial-based sensor locations and to report target position for fusion with other sensor.

g. *Satellite Coordinate System*. The coordinate frame with respect to which the sensor pointing requirements are derived. This system moves with the satellite. When multiple spacecraft are used to determine target position, each spacecraft has its own coordinate frame.

h. *Topocentric Horizon Coordinate System*. This system is typically used to represent the location of a target with respect to a terrestrial sensor system and is usually translated into Cartesian coordinates from azimuth and elevation angles.

3. OVERALL APPROACH TO THE PROBLEM

The goal of this research is to develop a methodology to improve the target tracking capability of a space-based sensor through dynamic bias correction. The combined tracking and error correction algorithm will 1) measure and estimate the sensor bias, 2) correct this bias, 3) improve track estimation quality, and 4) provide a comprehensive covariance of the target state estimate. This chapter develops our approach to this problem. As a sensor tracks a target against the cold background of space, the narrow-view tracking sensor will occasionally detect background stars in its field of view. These stars can be used to measure the observation error associated with the sensor. This error measurement can then be used to calibrate the sensor while tracking is occurring to improve the target state estimate.

3.1. Bias correction algorithm development

Figure 5 presents a nominal architecture to accomplish improved target tracking through bias correction and nonlinear state estimation. The space based IR tracking system comprises one or more tracking spacecraft and a ground processing system. The individual spacecraft provides distributed sensor readings to a centralized bias correction and state estimation processing system on the ground. A focal plane array captures infrared energy and an onboard processor performs pre-processing through threshold detection. The signals that exceed the threshold are reported to the ground processor in an object sighting message (OSM). The satellites also provide other data required to determine target location. This data includes spacecraft position determined by onboard Global Positioning System GPS and inertial navigation system, the spacecraft attitude, and timing data. The ground processor uses this information to estimate the sensor error and target position.

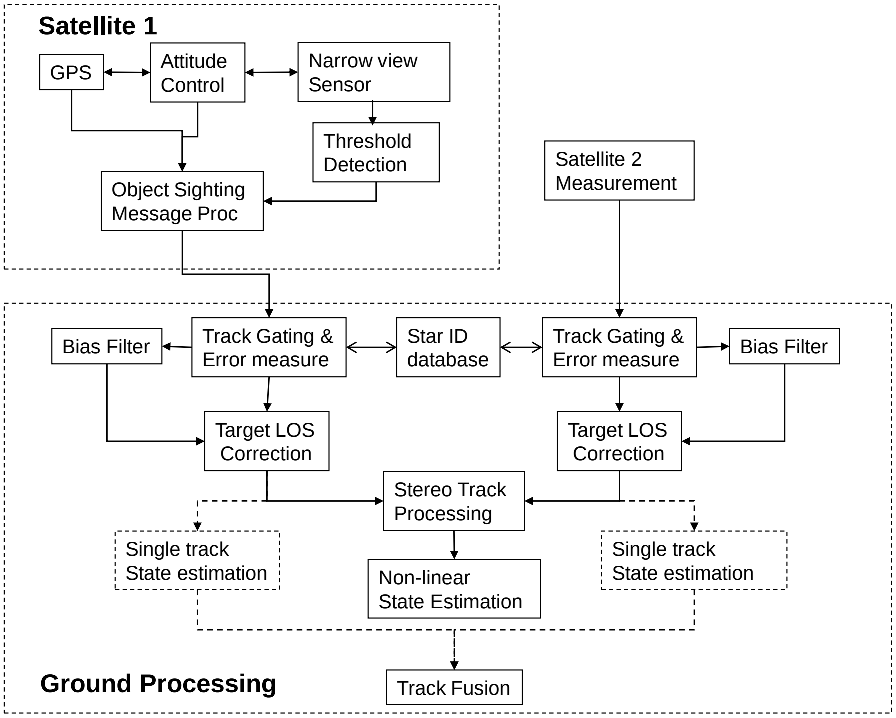


Figure 5. Nominal target estimation and tracking architecture

The system designed to accomplish this goal performs the following steps:

1. *Obtain Object Sighting Messages (OSM) from the sensor*. The sensor system or a pre-processing system on board the sensing satellite performs pre-analysis of the sensor detections and eliminates detections that do not meet threshold requirements. Detections of the target and other light sources that meet this

threshold are provided along with spacecraft attitude and position in an OSM to the ground station.

2. *Determine background star observations*. Prior to receiving a new object sighting message, the ground processor will attempt to determine if a star will be within the narrow sensor field of view. Highly accurate star maps consisting of tens of thousands of stars are available for use by the attitude control system [33]. The track processing system utilizes this catalog of stars within a star map database that identifies each star's magnitude and absolute location relative to the inertial reference coordinate system. If needed, star intensity is available to differentiate between stars or other objects.
3. *Measure and estimate sensor bias.* These true star positions are compared against the observations captured by the tracking sensor. If a star is expected, the processor will associate one of the objects reported in the object sighting message. The difference in the expected vector to the star and the measured observation represents the sensor bias. Correlations between stars and observations are made using a simple proximity association between the expected bearing and the star location. Adjacent stars should not cause confusion since the separation between the stars in the database is expected to be greater than the maximum sensor bias.
4. *Correct observation vector.* The individual sensor bias error observations are used to estimate the sensor bias. Two methods estimate sensor bias were tested; a sample and hold technique that used the last measured bias to correct the LOS

vector until a new measurement is made, and an Extended Kalman Filter (EKF) approach that attempts to provide an estimate of the bias in between bias measurements. If no star is detected in the field of view during a particular time step the last bias estimate is used until the next star is sighted when the bias error estimate can be updated. The bias error estimate is then used to correct the LOS vector to determine target location.

5. *Determine target location.* By combining target bearing with corrected LOS vector from another space sensor we can determine target locations. With the exception of the LKF that uses a intersection of the two observation vectors to determine final target position estimate, independent target estimates from the two individual sensors are fused to provide a refined estimate.
6. *Post correction state estimation*. As further observations are made the stereo track processor uses nonlinear estimation filter to propagate the target covariance matrix and estimated state vector. Following correction of the sensor observations of the target a nonlinear filter is used to determine the estimate of the target state and covariance. In this study we compared three different methodologies; a LKF, an EKF, and an UKF. A covariance matrix is also calculated as part of the state vector and compared to the Posterior Cramer Rho Lower Bound (PCRLB). This corrected target estimate is fed back to the spacecrafts for adjusting the sensor tracking calculations.

As shown in Figure 5, each spacecraft sends an object sighting message to the ground processor. The flow chart in Figure 6 breaks down the steps involved in the target tracking algorithm. These steps are those performed in the ground processing system described above. The portions within the dashed lines are the steps involved in the approach. As mentioned earlier, new track initiation is not considered in the research.

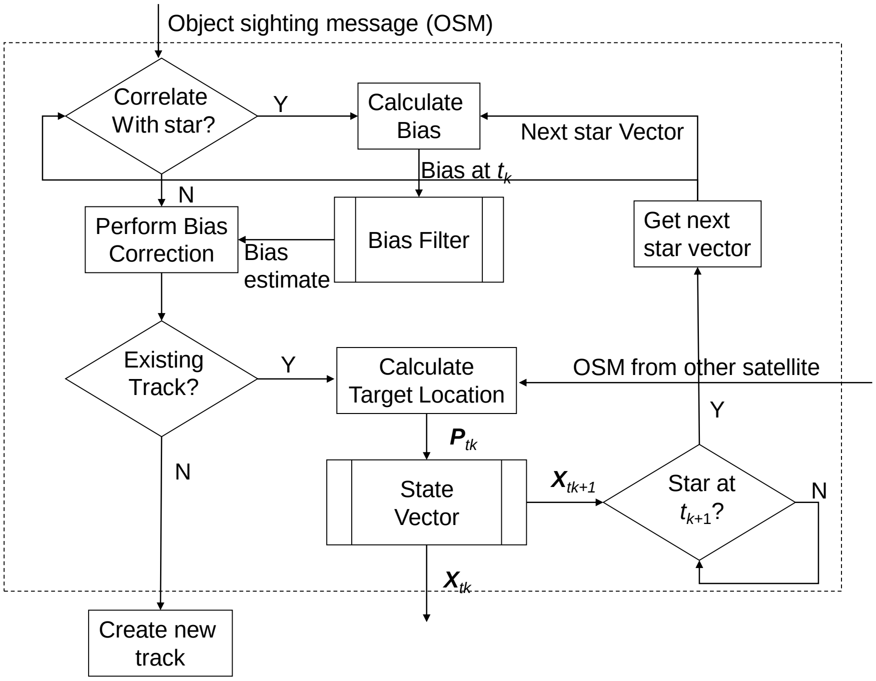


Figure 6. Bias Correction and Tracking Algorithm Flow Chart

3.2. Simulation modeling

The development and testing of dynamic bias correction and tracking algorithm proceeded in four phases. Phase I tested the effect of bias on the space tracking system. Phase II included testing of the star detection and sorting algorithm. The third phase compared the performance of the three nonlinear estimation filters following bias correction. Additionally, we tested the estimator's performance against the Cramer-Rao lower bound (CLRB) and check for filter consistency. Finally, in Phase IV we tested the bias correction and tracking algorithm against a number of different long and short range missile scenarios and a low earth orbit satellite using two different sensor satellite trajectories.

3.2.1. Satellite Tool Kit®

The missile and satellite trajectories for all scenarios were generated using the Satellite Tool Kit® (STK) [34]. The STK is a MATLAB based simulator produced by Analytical Graphics, Inc. (AGI) that models satellite and missile orbits and trajectories to a very high fidelity. The software provides users with the tools they need to perform such analysis with accurate modeling and visualization of missile defense systems. Figures 7 and 8 include screen shots of the STK application that displays some of the sensor satellites and ballistic missile targets used in this dissertation and a depiction of the scenario in three dimensions. There are other display capabilities showing tabular and graphical data associated with the dynamics of the sensor/target system. The STK software enables the following missile defense capabilities of the software [34]:

- Analyze simultaneous positions, relative geometries, and attitude motion
- Process post-burnout tracking data to predict trajectory, re-entry, and target coordinates
- Measure proposed systems with physics-based tracking data simulator
- Calculate missile intercept points
- Design custom missiles
- Perform automated trade studies

- Parametrically examine hundreds of missile trajectories reporting detection and tracking times from radars and other sensors.
-

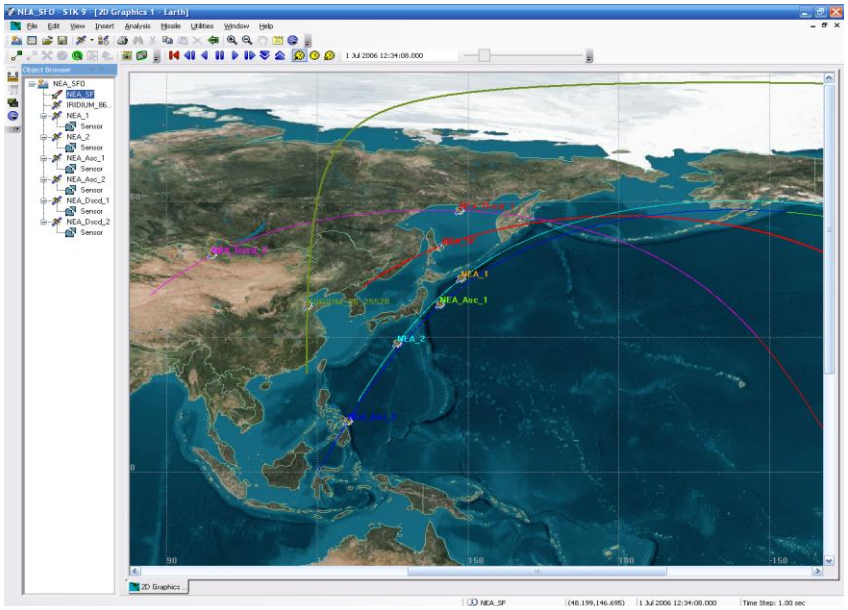


Figure 7. Screen shot of STK test scenario.

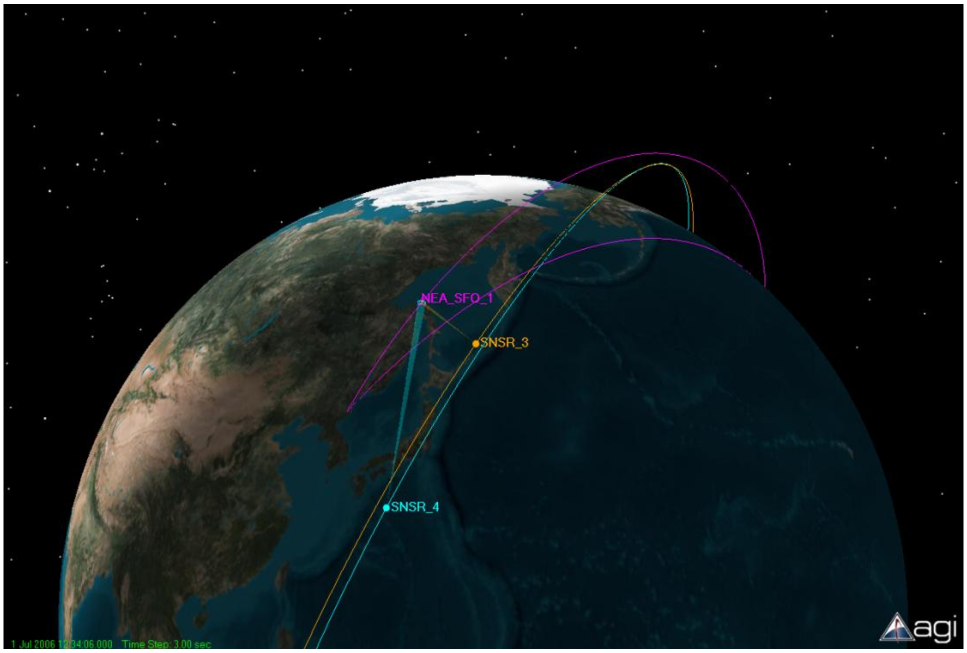


Figure 8. STK 3D display of tracking scenario.

3.2.2. Target model

For this study we used a predetermined scenario to aid in algorithm development and testing. This test scenario the target missile model represented a long range ballistic missile launched from Northeast Asia toward a target at the west coast of the United States. The STK simulation provided ECI reference frame track data for the target missile. The missile trajectory was adjusted to provide a flight time of approximately 30 minutes. The missile trajectory is displayed as the solid black line in Figure 9. A number of other threat scenarios were used for final evaluation and performance analysis and are explained in Chapter 6 and shown in Appendix A.

3.2.3. Satellite and sensor model

The STK model provided satellite position data, attitude data (quaternions), and sensor to target azimuth and elevation target observation measurements for two satellites and their IR sensors. The observation measurements were in reference to the satellite's instantaneous body frame attitude and were converted to the ECI frame using the satellite attitude quaternion measurements. The quaternion describes the orientation between the ECI frame and the satellite's current internal attitude reference (IAR). The two sensor satellites have a circular orbit of 600km altitude and 60 degree inclination. The constellation is a lead-follower configuration with the trail satellite about 5 minutes behind. The target launch time was selected so that the satellite sensors were able to follow the missile trajectory throughout most of its flight path. These satellite orbits enabled maximum visibility of the missile trajectory from multiple angles. As shown in Figure 9, as the sensor satellites cross the target track they experience constantly changing ranges and bearings to the target, and view the target from multiple angles, thus the effect of multiple scenarios are included in this one instance. Like the target model a number of satellite configurations and orbits were used for final evaluation and performance analysis and are explained in Chapter 6 and shown in Appendix A.

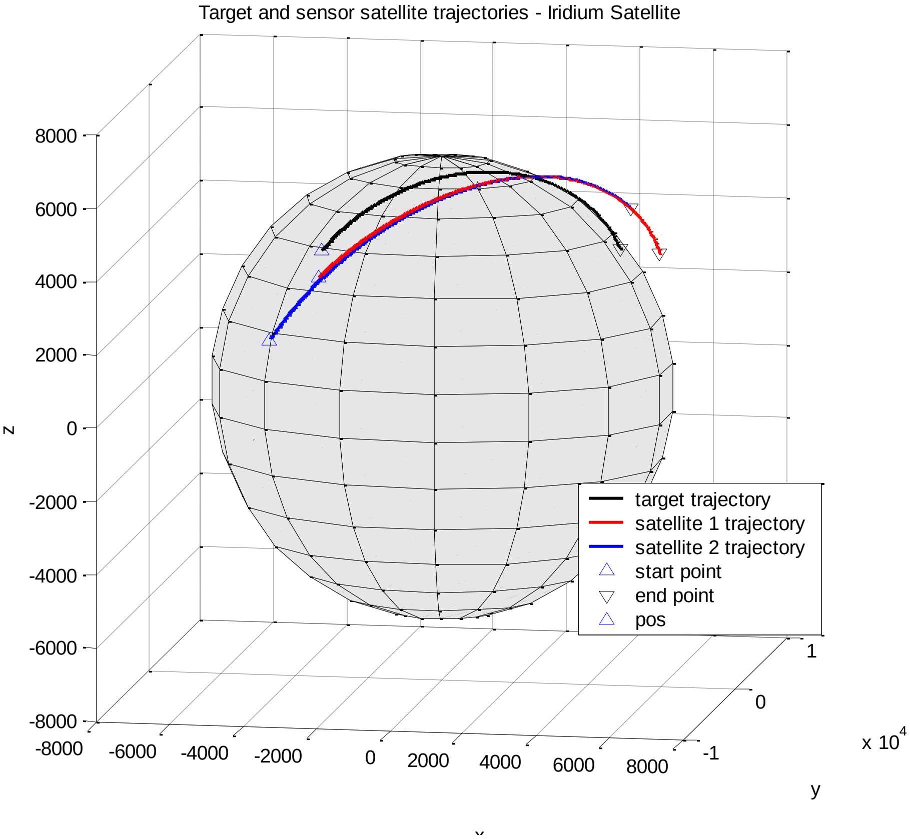


Figure 9. Target and satellite trajectories

4. BIAS MEASUREMENT AND CORRECTION

In this chapter we develop the theory and processes to use background stars in correcting bias error. We begin with an overview of how bias effects target tracking using a space based IR sensor. We then review how the IR sensor measures objects within its field of view. In the third section we develop an algorithm to quickly determine if a star is observable during an observation measurement epoch. If there is an observable star, we show how that star observation is transposed into the sensor reference frame to measure the bias. Next we use two bias models that represent the theoretical observation bias and propose two methods of bias estimation, a simple sample and hold technique, and an EKF bias estimator. We conclude the chapter with a comparison of both techniques to determine how well the celestial measurements correct the bias.

4.1. Effect of sensor bias on target tracking

To determine the effect of sensor bias on target tracking we made a number of simulation runs using various values for process, sensor, and bias error [35]. Table 3 provides a summary of the variables explored in the simulation.

Table 3. Research Variables

Variable	Minimum	Middle	Maximum
Azimuth Bias	.02 mrad	.2 mrad	2 mrad
Sensor error (σ_s)	5 μrad	10 μrad	20 μrad
Process error	10 m		100 m
Time Bias	0 sec	10 ms	
Target range	650 km		3000 km

The affect on target location error due to observation bias error and timing bias error were shown in Figure 2 of the introduction. In this early version of testing we used a Linearized Kalman Filter (LKF) as a means of improving target position estimates after target position was found via observation vector intercepts. The LKF is fully explained in the next chapter and is only used here as a starting reference. Table 4 below provides a comparison of mean error measurements of target position for various values of bias and filter sensor variance for a process noise of 10 meters standard deviation (STD). Results are shown for both the LKF and when the intercept position taken without filtering (represented by the No filter columns in the table).

The most significant result is the inability of the LKF to overcome the effect of bias error. As shown in the columns for σ_s = 20 μrad and 1 *m*rad, the filter only provides an improvement over the unfiltered observations when the filter variance error is of the same magnitude, or greater than, the bias error. For example, with σ_s = 20 μrad, and bias = 200 μrad, we have an improvement in target error to 233.6 meters from 377.5 meters. This is expected since the sensor error in the LKF calculation for the Kalman gain is dominated by the inaccuracy of the measurement due to the bias error. When the filter sensor variance is the same or greater than the bias error, the sensor variance subsumes the bias error in the observations and is dominant in driving the target error. In other words, the filter assumes the sensor provides a better measurement than is actually possible.

Table 4. Mean error in target position for various sensor noise values.

σ_p=10 m	$\sigma_s = 5\ \mu$rad		$\sigma_s = 10\ \mu$rad		$\sigma_s = 20\ \mu$rad		σ_s = 1*m*rad	
Bias	No filter	LKF	No filter	LKF	No filter	LKF	No filter	LKF
None	21.6 m	19.2 m	33.8 m	28.1 m	62.2 m	42.5 m	3.01 km	1.51 km
.02 *m*rad	42.5 m	41.9 m	49.6 m	47.1 m	72.6 m	48.8 m	3.02 km	1.51 km
.2 *m*rad	373.6 m	373.8 m	374.5 m	374.9 m	377.5 m	233.6 m	3.03 km	1.57 km
2 *m*rad	3.73 km	3.73 km	3.73 km	3.73 km	3.73 km	3.73 km	4.72 km	4.08 km

To explore the effect of timing error, we introduced a 10 *m*s delay in the first satellite's observation time. This result is shown along with two values for process noise STD in Table 5. Here we see that process noise has little effect on the filter performance in cases of large bias error. When process noise is on the order of 100 meters, the LKF provides no improvement over the intercept measurements. However, timing bias error increases the error for both the filtered and unfiltered results. Essentially, the timing bias has the same effect as an increased measurement angle bias.

Table 5. Mean error in target position for constant sensor error

$\sigma_s = 10\ \mu$rad	σ_p = 10 m		σ_p = 100 m		Δt = 10 ms	
Sensor Bias	No filter	LKF	No filter	LKF	No filter	LKF
None	33.8 m	28.1 m	33.8 m	33.1 m	61.6 m	60.1 m
.02 *m*rad	49.6 m	47.1 m	49.5 m	49.2 m	66.1 m	64.7 m
.2 *m*rad	374.5 m	374.9 m	374.4 m	374.4 m	383.7 m	384.6 m
2 *m*rad	3.73 km	3.73 km	3.73 km	3.73 km	3.74 km	3.74 km

We also found that bias in different sensor planes had no major effect on target estimation. As seen in Figure 10, target position error remains at the same order of magnitude when bias is modeled in the azimuth and elevation measurements for each satellite in turn. The major difference between scenarios is how geometrical configuration of satellite position and orientation with respect to the target affect the shape of the error curves over time. This effect also would be reflected in different satellite/target orientations.

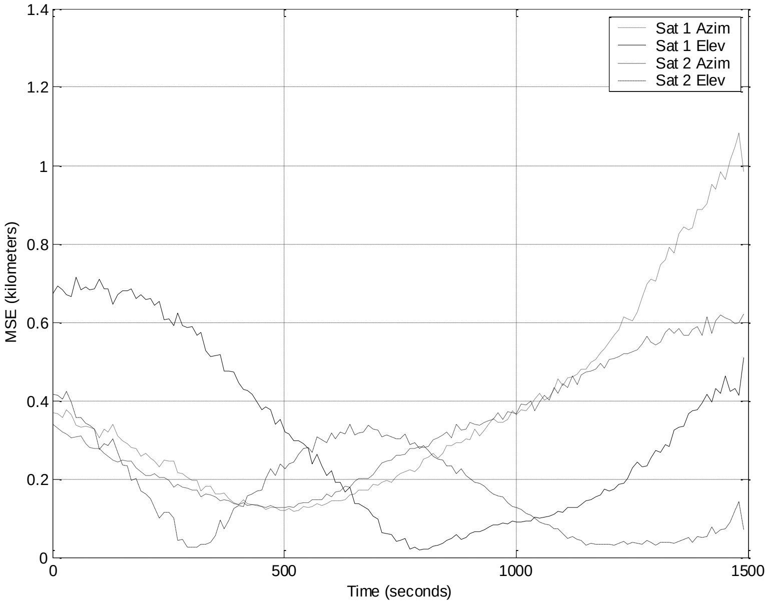


Figure 10. Mean error of target position for .2 *m*rads bias error in azimuth and elevation for each sensor.

4.2. Measuring sensor line of sight

The IR sensor consists of an optical system and a planar measurement device. Optical measurement is normally expressed in terms of elevation and azimuth to the target from some sensor defined reference point. The light from the target and stars is focused on the detector plane. The focal plane and the sensor bore sight make up the sensor coordinate frame (CS_k) as shown in Figure 11. The *x*-axis points along the sensor bore sight vector and has a focal length *F*. The *z*-axis points along the gimble elevation rotation axis and measures azimuth distance, and the *y*-axis points along the gimble azimuth rotation axis and measures elevation.

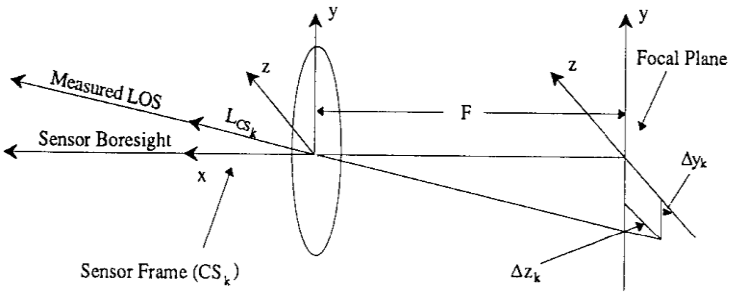


Figure 11. Optical sensor measurement

The measured target LOS at time t_k is represented by a vector $\bar{L}_{Csk}$ defined as [36];

$$\vec{L}_{Csk} = \left[1 \quad -{}^{\Delta y_k}/_{F} \quad -{}^{\Delta z_k}/_{F}\right]^T, \tag{4-1}$$

and the unit vector of the sensor boresight is

$$l = [sin\theta_k cos\phi_k \quad cos\theta_k cos\phi_k \quad sin\phi_k], \tag{4-2}$$

where θ_k and ϕ_k are the azimuth and elevation of the sensor in the satellite's body frame, respectively.

In this study we assume the use of subpixel centrioding as a methodology to increase the accuracy of a detected object in a photoelectronic array. An object viewed by the sensor will occupy several pixels. Once a pixel is detected, a region of interest window is aligned with the detected pixel in the center. An intensity value of all the pixels surrounding the pixel(s) with peak intensity is measured. The centroid and brightness are then calculated from the averaging of these pixel values [37].

4.3. Determining background star observations

Since stars are extremely distant from Earth they appear fixed from the Earth within the ECI reference frame. Thus, they have long served as a stable reference for spacecraft attitude control [38] and sensor calibration in high altitude systems [39]. For attitude control star sensors periodically measure the deflection of a number of stars from their expected positions as provided by the spacecraft's attitude control system. This measurement consists of both rotational and directional information and provides an input into the attitude control algorithms. Here, we discuss how we can use the same information to correct sensor bias.

4.3.1. Star catalog and star detection

A star map catalog, called SKYMAP, containing the precise location and magnitudes of over 300,000 stars are available from NASA for use in spacecraft attitude control. Various satellite missions use portions of the catalog for their attitude control star database. We choose to use one of these databases for this study; specifically our database is derived from the SKYMAP Sub-catalog for the star sensors aboard NASA's Aura Constellation satellites. The Aura database contains over 25,000 stars with magnitudes above +7.25 [40]. These stars are distributed randomly across the celestial sphere. Stars do move in relation to the each other and the earth, but this movement is known and is compensated by applying a correction factor that is also available in the star catalog. The earth's gravitational field does not have an appreciable effect on the starlight and is discounted.

For this study, the Aura catalog was modified to include only the x, y, z components of the star location and then normalized to present a unit vector. Although the star magnitude is available for helping with star identification, it was not used in this study. Another method of identifying a stars' location is through its right ascension and declination. Right ascension is a measurement clockwise around the equator (0 to 360 degrees) of the celestial sphere from the First Point of Aires, which defines the x-axis in the celestial sphere. Declination is measured in degrees above and below the celestial equator (-90 to 90 degrees). These values are found through a standard Cartesian to Spherical conversion. Figure 12 is a graphical representation of star positions represented by right ascensions and declination from the Aura star database for a 20 degree by 10 degree area of the celestial sphere.

A critical issue in using celestial observations for sensor calibration is to determine whether a star will appear in the narrow view sensors field of view while it is tracking the target, and whether there will be enough star observations during the tracking period to estimate the sensor bias. The double line through the background represents a sensor with a 0.5 degree field of view. As seen, there are a number of opportunities for the sensor to detect background stars and make measurements that can be used to make correct the sensor bias.

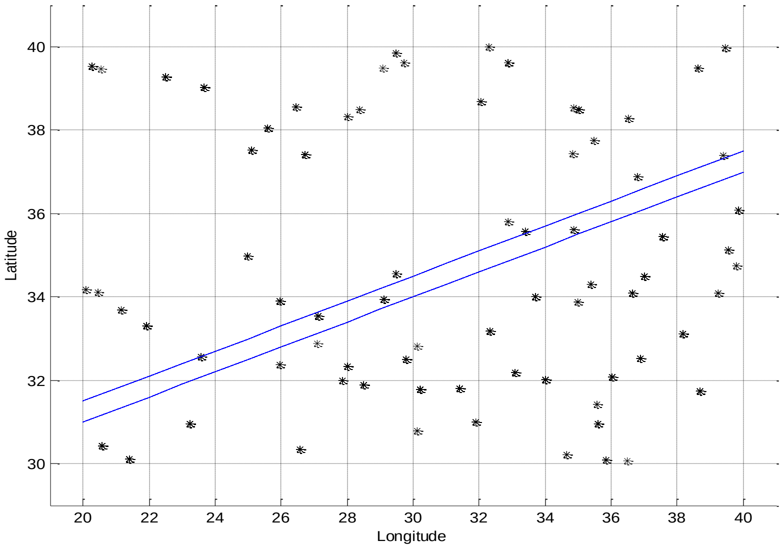


Figure 12. Star map of 20 x 10 degree portion of the sky.

4.3.2. Rapid star identification algorithm

As seen in Figure 12, only a small number of the stars within the catalog of stars will be near the sensor line of sight as it projects a track through the space background. Therefore, we do not need to search the entire star catalog to find the stars that are candidates to be within this sensor field of view. Since we anticipate using star observations about once a second for bias correction we could not afford to perform a real-time search of the entire star catalog. Therefore, we developed a search algorithm to speed up the star identification [41] and [42].

We must first determine if the sensors' line of sight to the star is blocked by the earth. We accomplish this by calculating the angle between the satellite position vector in reference to the center of the earth (the negative component of the position in the ECI reference frame normalized into a unit vector) and the earth limb based on the current position of the satellite. This angle is compared to the angle between the earth vector and the sensor LOS. Figure 13 shows a diagram of the geometry involved. Here we define v_e as the satellites position vector and then with a dot product between the vectors we have

$$\theta_{los} = \cos^{-1}(v_e \cdot l), \tag{4-3}$$

and

$$\theta_e = \sin^{-1}\left({}^{R_e}\!/_{alt}\right), \tag{4-4}$$

where R_e is the radius of the earth and alt is the altitude of the satellite. If θ_{los} is greater than θ_e, a star may be observed. The same holds true for an attempt to observe the target, although, at that range, the target is probably too dim to be seen.

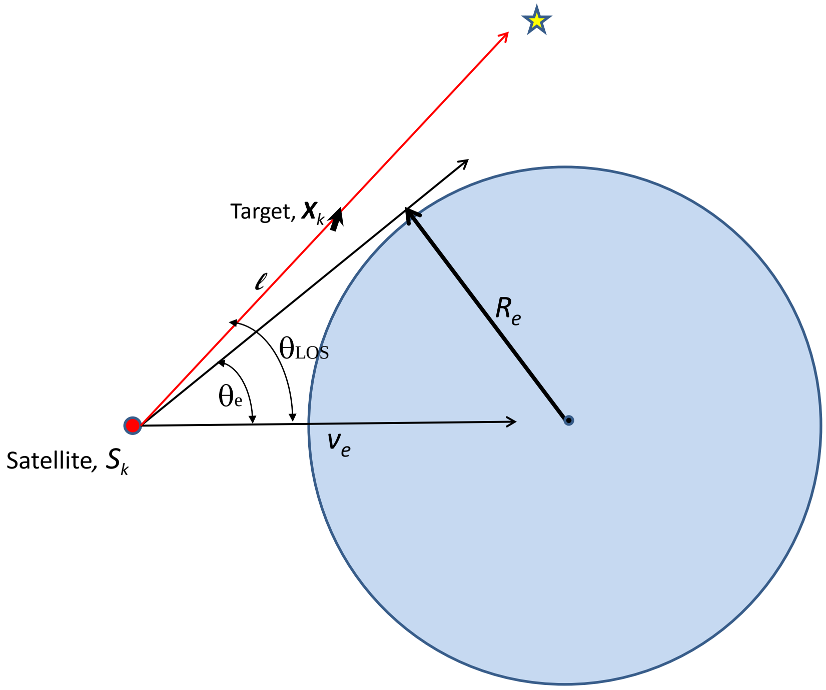


Figure 13. Star observation geometry

During the initial observation the entire star catalog is compared to the sensor line of sight. Following identification of the star nearest the LOS in the next time step each star in the nearest neighbor table is compared with the sensor LOS vector to determine if the star is within view. If the star is within view, its number is stored for catalog lookup during the error measurement step. When a different star is identified as the nearest to the LOS it becomes the entry into the nearest neighbor table to begin the next search.

To more rapidly determine if a star was within a sensor's field of view we constructed a table of each star and its 11 closest neighbors off-line. A portion of this table is shown in Table 6 where each number represents a unique star ID. This method makes use of the knowledge from a previous time step to initiate the current time star search [43]. At the beginning of narrow band tracking the entire star catalog is compared to the sensor line of sight. In the next time step only the stars in the nearest neighbor table associated with the previous nearest star are compared with the sensor LOS vector to determine if the star is within view. If the star is within view, its number is stored for catalog lookup during the next error measurement step. When a different star is identified as the nearest to the LOS it becomes the entry into the nearest neighbor table to begin the next search.

Table 6. Portion of nearest neighbor star catalog

Star #	1	2	3	4	5	6	7	8	9	10	11
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
8637	8547	8704	8725	8497	8514	8499	8707	8634	8739	8438	8707
8638	8575	8554	8501	8709	8633	8691	8699	8795	8604	8733	8699
8639	8450	8669	8673	8732	8656	8504	8619	8653	8539	8886	8619
8640	8692	8807	8560	8528	8508	8503	8659	8586	8805	8624	8659
8641	8752	8660	8696	8690	8736	8492	8583	8603	8563	8705	8583
8642	8652	8470	8722	8616	8406	8786	8494	8814	8866	8286	8494
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

As target tracking proceeds there will be periods when stars are not within the field of view. If no star is detected within the field of view over a 60 second period, the algorithm resorts to an entire table search again to find the star nearest the line of sight. This is important when the sensor is blocked by the earth or moves rapidly across the sky as the sensor LOS can move away from the previously calculated 11 closest stars. Figure 14 shows a flow diagram of the algorithm.

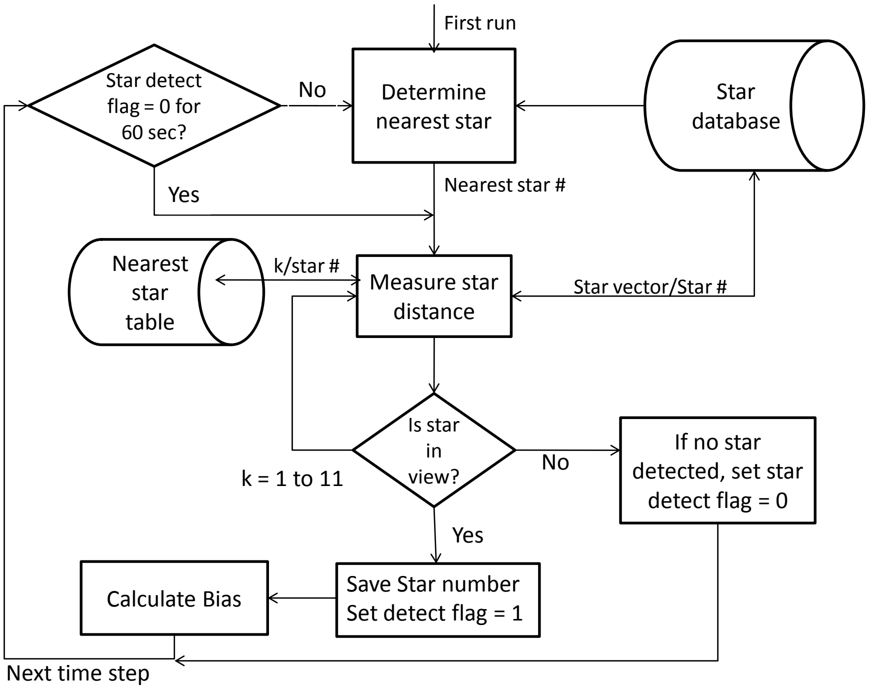


Figure 14. Star detection algorithm

4.3.3. Star vector rotation and field of view measurement

The known star vector is rotated from the ECI frame onto the sensor frame by using the satellite's attitude quaternion. For example, a rotation about the unit vector $\overline{n}$ by an angle θ is computed using the quaternion

$$q = \left(\cos\left({}^{\theta}/_{2}\right), \sin\left({}^{\theta}/_{2}\right)\right) = q_w + iq_x + jq_y + kq_z\ , \tag{4-5}$$

which can be used to describe a rotation matrix,

$$C_{ECI}^{IMU} = \begin{bmatrix} 1 - 2q_y^2 - 2q_z^2 & 2q_xq_y - 2q_zq_w & 2q_xq_z + 2q_yq_w \\ 2q_xq_y + 2q_zq_w & 1 - 2q_x^2 - 2q_z^2 & 2q_yq_z + 2q_xq_w \\ 2q_xq_z - 2q_yq_w & 2q_yq_z - 2q_xq_w & 1 - 2q_x^2 - 2q_y^2 \end{bmatrix}. \tag{4-6}$$

We can then rotate the star vector from ECI to body frame as $\bar{s}_{IMU} = C_{ECI}^{IMU}\,\bar{s}_{ECI}$, which can be further rotated to the sensor frame via the sensor azimuth and elevation from

$$C_{IMU}^{Sensor} = \begin{bmatrix} \cos\theta_s \cos\phi_s \\ \sin\theta_s \cos\phi_s \\ \sin\phi_s \end{bmatrix}, \tag{4-7}$$

where θ_s and ϕ_s are the sensor azimuth and elevation measurements respectively. Therefore, the star vector rotation to the sensor frame can be found from

$$\bar{s}_{sensor} = C_{IMU}^{sensor}\, C_{ECI}^{IMU}\, \bar{s}_{ECI}. \tag{4-8}$$

We determine if a star will be within the field of view by taking the dot product of each star vector and the anticipated sensor LOS vector, namely,

$$\theta_{star} = \cos^{-1}\left(\vec{s} \cdot \vec{l}\right), \tag{4-9}$$

where ***s*** is the vector of the star position in the sensor coordinate frame, and ***l*** is the sensor vector of the sensor LOS. If the angle θ_{star} is within the sensor field of view the detector is assumed to be able to see the star.

For example, Figure 15 presents a scenario where two IR sensor trajectories and the respective detected stars are shown. The arrows represent the direction of sensor movement. In this scenario, the sensor aboard Satellite 1 made 357 star detections and the sensor aboard Satellite 2 made 836 star detections in 1500 seconds of tracking time with about a 0.25 and 0.5 Hertz detection rate respectively. The narrow band sensors do not track the target during the first 300 seconds of flight. This is the period where the wide-band sensor is tracking the target which is too bright for the narrow view sensor. The sensor field of view in this scenario is one degree in diameter. In comparing the two sensor trajectories, one can see that the number of stars detected is partially determined by the length of the path traced by the sensor. Sensor 1's trajectory is about two thirds the length of sensor 2. However, in some cases, the same star will remain in the sensor field of view for a number of time steps. This will typically occur when the sensor trace moves slowly across the celestial sphere.

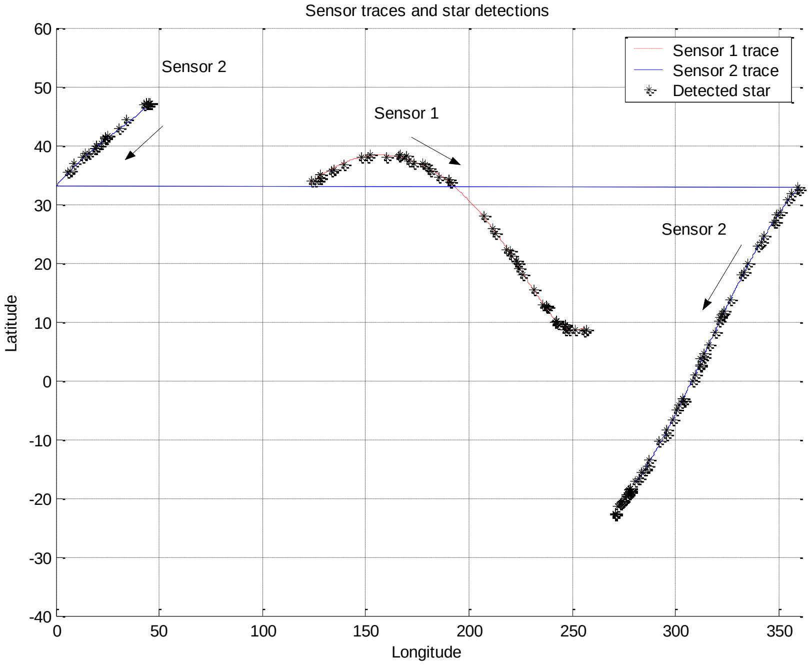


Figure 15. Sensor trajectories with detected stars.

4.3.4. Effect of aperture size

To determine the effect of the number of stars observed on the accuracy of the target track we measured the number of stars versus the sensor aperture. One can see from Table 7 and Figure 16 that as the aperture decreased the number of detected stars decreased, and correspondingly, the mean error also increased almost linearly with the sensor field of view. The method of measuring mean error is discussed in detail later.

Table 7. Mean error and number of stars with respect to aperture size

Aperture size (diameter in degrees)	Stars seen by Sensor 1	Stars seen by Sensor 2	Mean Error (EKF with LKF)
.6	102	306	56.4 m
.8	210	510	43.4 m
1.0	328	799	28.8 m

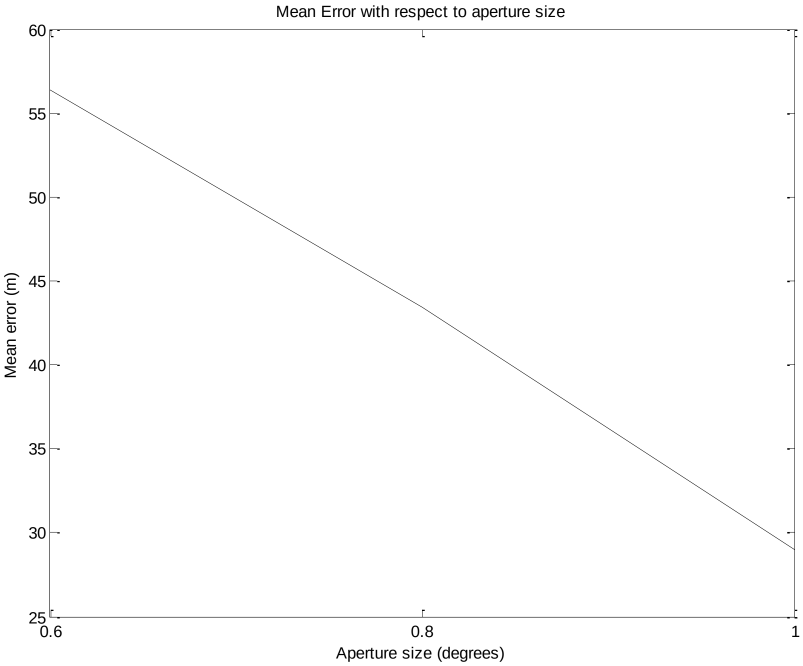


Figure 16. Mean error with respect to aperture size

4.4. Bias modeling and measurement

Sensor bias can be measured as the difference of the expected LOS and the actual LOS in the z-y plane. Each star sighting allows comparison with cataloged right ascension and declination angles with payload line of sight measurements to produce celestial residuals that are input to a filter to produce an estimate of the bias in the sensor pointing vector. This bias can be removed from the target line of sight vector to produce a corrected target observation.

4.4.1. Bias model

As discussed by Kristosturian in [31], a satellite undergoes a cyclical heating and cooling due to its orbit around the earth and most sensor bias error is due to the effects of thermal distortions as the satellite enters and comes out of Earth eclipse. This cycle has a period of about 90 to 120 minutes. In his paper Kristosturian provides reasonable characterization of the bias as

$$\psi = \begin{pmatrix} \psi_x \\ \psi_y \\ \psi_z \end{pmatrix} = \begin{pmatrix} b_x + S_x \sin\frac{2\pi}{T}(t-\varphi) + \eta_x \\ b_y + S_y \sin\frac{2\pi}{T}(t-\varphi) + \eta_y \\ b_z + S_z \sin\frac{2\pi}{T}(t-\varphi) + \eta_z \end{pmatrix} \tag{4-10}$$

where

b_x, b_y, b_z are the indeterminate angular components of ψ considered to be normally distributed random constants, and

S_x, S_y, S_z are the indeterminate amplitudes of the sinusoidal varying components of ψ induced by thermal variation defined by period T and phase φ. They are also assumed to be normally distributed, random constants.

η_x, η_y, η_z = random disturbance components that are assumed to be zero-mean normally distributed, random variables each with known variance.

If we translate this Cartesian representation into an azimuth and elevation polar form we can describe the bias associated with the circular orbit of the sensor satellites as constant bias with a sinusoidal changing component that can be represented by

$$\beta(t) = \beta_s + \beta_0 \cos(\varpi t + \varphi) + v(t) \tag{4-11}$$

where β_s is the steady state bias value, and β_0 is the amplitude of the sinusoidal component with a constant or near constant frequency ω and phase φ. This bias is present in both azimuth and elevation, and since they are orthogonal, the phase difference between the components is 90 degrees. The zero mean Gaussian bias process noise $\nu(t)$ accounts for other random bias error sources and has a variance σ_β.

Taking the first and second derivatives of the bias we have

$$\dot{\beta}(t) = -\beta_0 \varpi \sin \varpi t, \tag{4-12}$$

and,

$$\ddot{\beta}(t) = -\beta_0 \varpi^2 \cos \varpi t = -\,\beta_0 \varpi^2 \beta(t). \tag{4-13}$$

4.4.2. Measuring sensor bias

To measure the instantaneous bias the expected star position is compared to the actual measured star position in the sensor frame to determine the azimuth and elevation bias components [41] and [42]. Figure 17 shows how these measurements are made. Given a sensor focal length F the difference in the expected position and the actual position is related to the measured bias as

$$\begin{aligned} \beta_{az} &= \tan^{-1}(\Delta z_{ex} - \Delta z_{act}/F) \\ \beta_{el} &= \tan^{-1}(\Delta z_{ex} - \Delta z_{act}/F) \end{aligned} \tag{4-14}$$

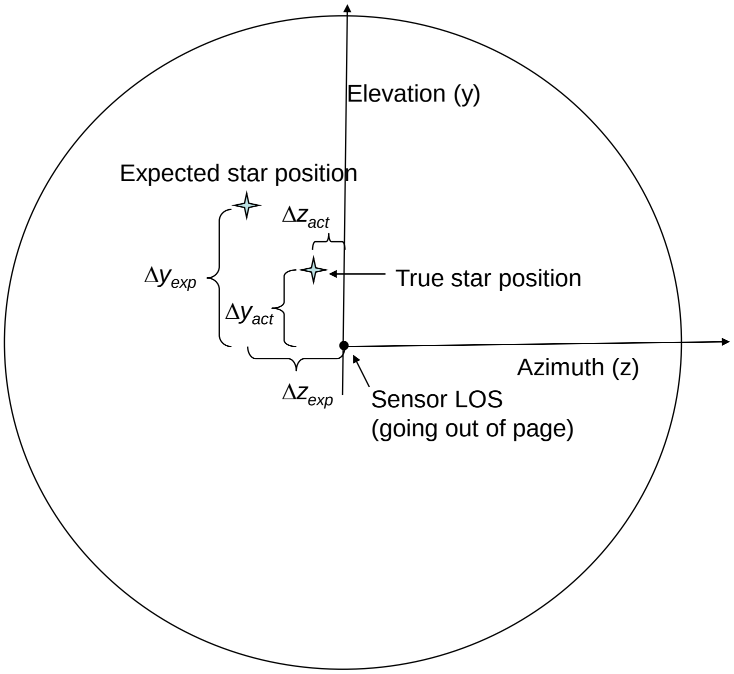


Figure 17. Bias measurement in sensor field of view

Sometimes there may be more than one star in the field of view that are in the catalog of stars. In these cases, all the known stars are measured and an average of the measurements is calculated to determine the bias for that time step.

4.4.3. Filtering the bias measurements

A three state EKF produces a refined estimate of the bias state for each sensor prior to target LOS vector correction. Since the bias assumed sinusoidal in nature [31], a bias state vector for each sensor axis describes bias magnitude, bias rate of change and the frequency of the sinusoidal in both azimuth and elevation in the sensor frame at each time step *k*, such that

$$\bar{\beta}_k = [\beta_k \quad \dot{\beta}_k \quad \varpi_k]. \tag{4-15}$$

The estimate of the bias state is determined by the standard EKF equation

$$\hat{\beta}_k = \mathrm{A}_\beta \hat{\beta}_{k-1} + K_\beta \left(z_{(\beta)k} - H_\beta \mathrm{A}_\beta \hat{\beta}_{k-1}\right), \tag{4-16}$$

where the fundamental matrix derived from the preceding differential equations (4-11), (4-12) and (4-13) can be approximated by [44]:

$$\Phi_\beta \approx \begin{bmatrix} 1 & T_s & 0 \\ -\hat{\varpi}_{k-1}^2 T_s & 1 & -2\hat{\varpi}_{k-1}\hat{\beta}_{k-1} T_s \\ 0 & 0 & 1 \end{bmatrix} \tag{4-17}$$

and the corresponding process noise matrix of

$$Q_\beta = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1.333\hat{\varpi}_{k-1}^2 T_s^3 \sigma_\beta^2 & -2\hat{\varpi}_{k-1}\hat{\beta}_{k-1} T_s^2 \sigma_\beta^2 \\ 0 & -2\hat{\varpi}_{k-1}\hat{\beta}_{k-1} T_s^2 \sigma_\beta^2 & T_s \sigma_\beta^2 \end{bmatrix} \tag{4-18}$$

where $\hat{\beta}$ and $\hat{\varpi}$ are the estimated bias and bias frequency, σ_β^2 is the variance of the bias process noise, and T_s is the time step interval of the measurements. Since star observations are not made at every time step the Kalman prediction will require taking the time difference into account so there may be a different T_s at each measurement.

The EKF uses only the observation of bias magnitude at times when stars are observed and the observation matrix H_β only accounts for the magnitude of the bias. Thus we have

$$z_{(\beta)k} = \beta_k, \quad \text{and} \quad H_\beta = [1 \quad 0 \quad 0]. \tag{4-19}$$

The standard EKF equations for the bias estimate are,

$$M = \mathrm{A}_\beta P_{(\beta)k-1} \mathrm{A}_\beta^T + Q_\beta$$

$$K_\beta = M H_\beta^T \left(H_\beta M H_\beta^T + R_\beta\right)^{-1} \tag{4-20}$$

and

$$\hat{\beta}_k = \tilde{\beta}_{k-1} + K_\beta\left(z_{(\beta)k} - \tilde{\beta}_{k-1}\right)$$

$$P_{(\beta)k} = \left(I - K_\beta H_\beta\right)M \tag{4-21}$$

where K_β is the Kalman gain, $R_\beta = \left[\sigma_\varsigma^2\right]$ is the bias measurement error variance, and $\tilde{\beta}$ is the projection of the state vector $\bar{\beta}$ calculated by numerical integration. Between star detections, the filter propagates the last to the next observation.

4.4.4. Target observation correction

Line of sight correction is performed by subtracting the bias estimate obtained above from the observed target line of sight as represented by the azimuth, $\tilde{\theta}_i$ and elevation, $\tilde{\phi}_i$. This correction is performed in the sensor frame and results in new azimuth and elevation measurements

$$\theta_i = \tilde{\theta}_i - \hat{\beta}_\theta$$

$$\phi_i = \tilde{\phi}_i - \hat{\beta}_\phi. \tag{4-22}$$

These new values are then used to estimate the target location.

4.5. Bias error scenario development and validation

To better test the bias estimation methodology, we define two cases of sensor bias error modeled by equation (4-11). The first case (shown in Figure 18) involves a purely sinusoidal bias error with $\beta_s = 0$ and $\beta_0 = .2$ *m*rads as given in eqn. (4-11) and is representative of the cyclical thermal effects of the satellite's orbit. The second case (Figure 19) represents a constant bias error with a small sinusoidal component, or $\beta_s = .2$ *m*rads and $\beta_0 = .02$ *m*rads. In this second case a constant bias is representative of a bias error due to misalignment of sensors. Each figure shows both the azimuth bias (upper curve) and the orthogonal elevation bias (lower curve) for one satellite. In the model, sensor error is added to the sensor measurements while they are still in the satellite body-oriented reference frame. This error includes both zero-mean Gaussian measurement noise, and the bias error.

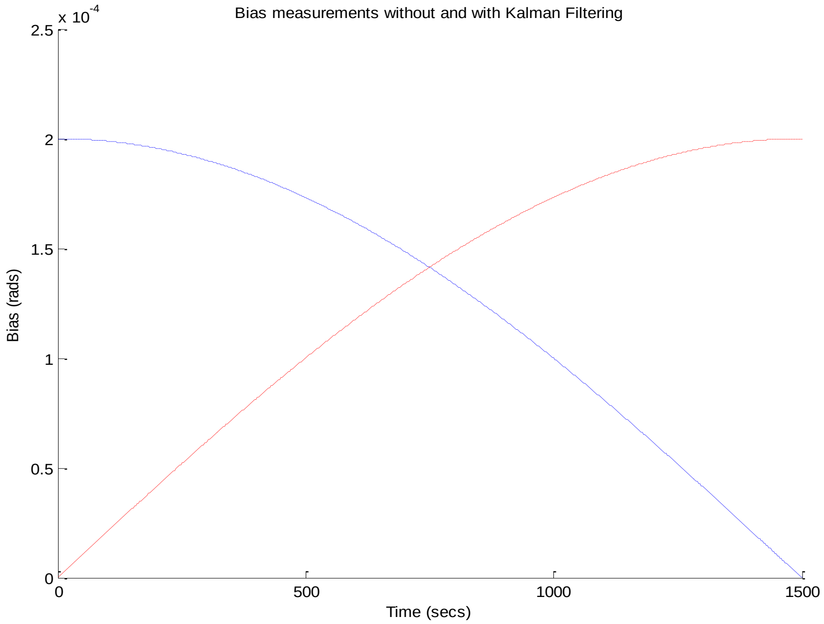


Figure 18. Case 1 - Pure sinusoidal bias model

Two methodologies for estimating the bias were compared – use of direct bias measurements without an EKF (simple bias correction) and derived bias estimates with an EKF. When using the first method (without the EKF) the last bias obtained from a star measurement was used as the measurement bias in the time interval between star observations to correct the target line of sight. The second methodology utilized the EKF developed in section 4.4.3 to predict the bias estimate at each time step.

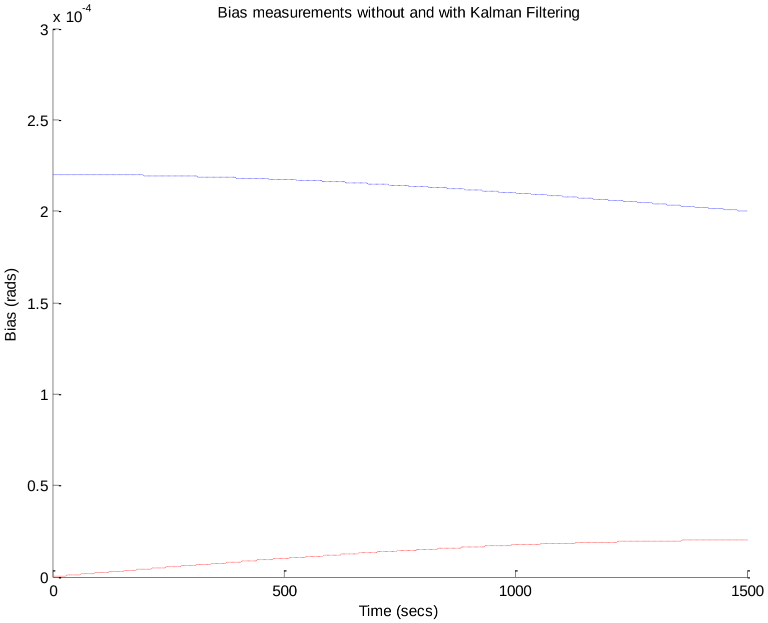


Figure 19. Case 2 - Constant bias with sinusoidal component model

Figures 20 and 21 show the measured bias compared to the filtered bias in both cases. In both plots the initial bias estimate is zero until a measurement is obtained when the sensors can begin detecting stars (t = 300 seconds). The dashed lines show the true sensor bias, the dotted lines show the simple bias measurement and the solid lines show the bias estimates from the EKF. The simple bias estimate diverges greatly from the true bias between star measurements for the sinusoidal bias, but does a reasonable job in the near-constant bias case. However, one can see that the EKF does well in estimating both bias trajectories. The filter was initialized with zero bias and zero bias rate, but with an initial estimate for the frequency based on *a priori* knowledge. In the mostly constant bias case, $\omega_0 = 0$, and for the pure sinusoidal case, $\omega_0 = 001$ [42].

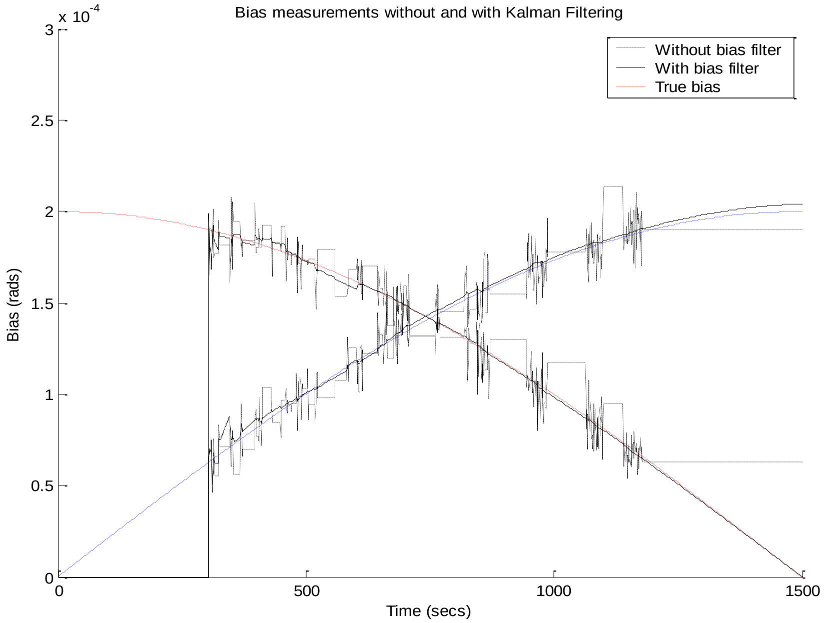


Figure 20. Case 1 - Bias measurements and estimates for pure sinusoidal bias

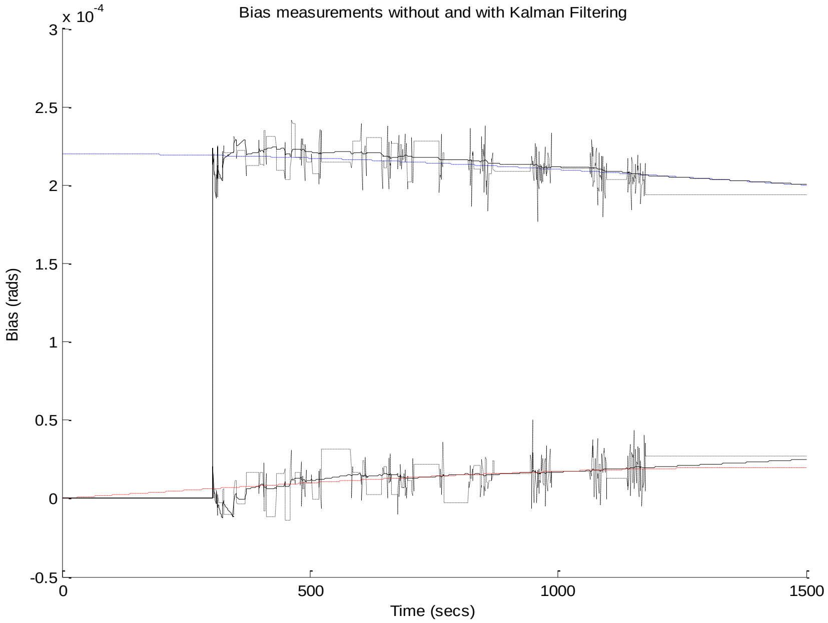


Figure 21. Case 2 - Bias measurements and estimates for constant bias with sinusoidal component

Using a test scenario we studied the effect on tracking accuracy for the two cases of bias error, corresponding to the bias waveforms shown in Figures 18 and 19 above was analyzed with and without a bias filter for 500 Monte Carlo runs. In the first case, bias error was modeled using a sinusoidal bias in both the azimuth and elevation axis of both satellites using equation (4-11) with $\beta_s = 0$ and $\beta_0 = 0.0$, 0.02, 0.2 and 2.0 *m*rads. The model assumed a target process noise of 10 meters STD, sensor observation noise of 10 μrads STD in both azimuth and elevation, and a bias process noise of 2 μrads STD.

Figure 22 illustrates the estimate error from the true target position for the $\beta_0 = 0.2$ *m*rads case using the bias correction algorithms developed earlier. One can see the significant reduction in position error when the sensor is able to begin detecting stars and correcting for bias error at $t = 300$ seconds into target flight. The error begins to increase near the end of the tracking period when stars are not detected by the sensor and the simple bias correction diverges from the true bias. Table 8 provides a comparison of mean error for various values of β_0 with and without bias corrections.

The second bias case uses the larger constant bias with a small sinusoidal overlay described in Figure 19. In this case $\beta_s = 0.0$, 0.02, 0.2 and 2.0 *m*rads, $\beta_0 = 0.1\beta_s$ for the azimuth measurement, and $\beta_s = 0$ for the elevation measurement. As in Figure 22, Figure 23 shows that the correction of the error is also significant. However, the simple bias does not diverge as much in this scenario since the bias changes little, but estimating the bias using the EKF still provides a more accurate tracking of the target. Table 9 provides a comparison similar to Table 8.

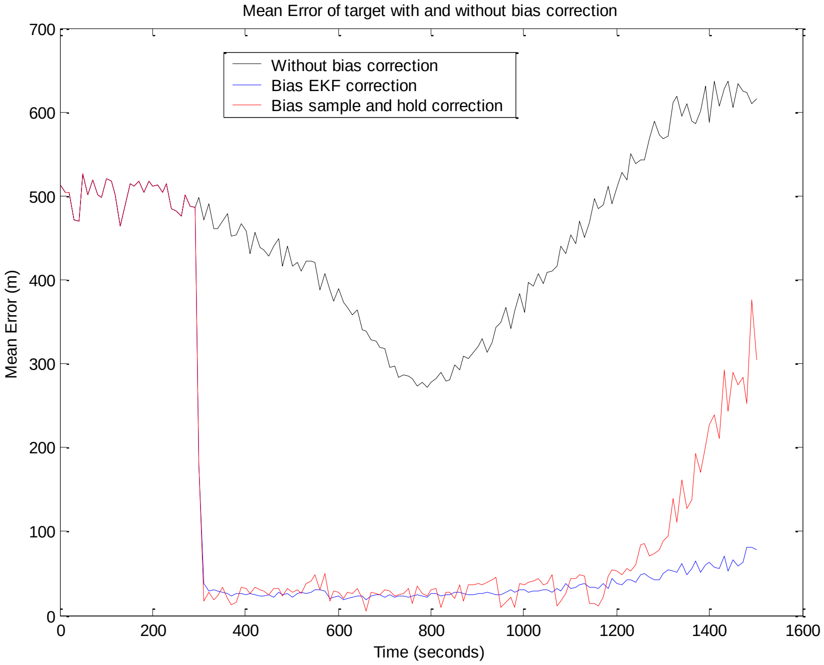


Figure 22. Case 1 - Mean error of target position for sinusoidal bias (β_0 = 0.2 *m*rads)

Table 8. Mean error in target position for pure sinusoidal bias (σ_s = 10 μrad, σ_p = 10 m, σ_β = 2 μrad, σ_ζ = 10 μrad)

Bias (β_0)	No bias correction	Simple bias correction	With EKF bias filter
None	40.6 m	47.7 m	41.0 m
.02 *m*rad	52.9 m	46.2 m	41.1 m
.2 *m*rad	489 m	65.3 m	41.8 m
2 *m*rad	4.88 km	374 m	55.3 m

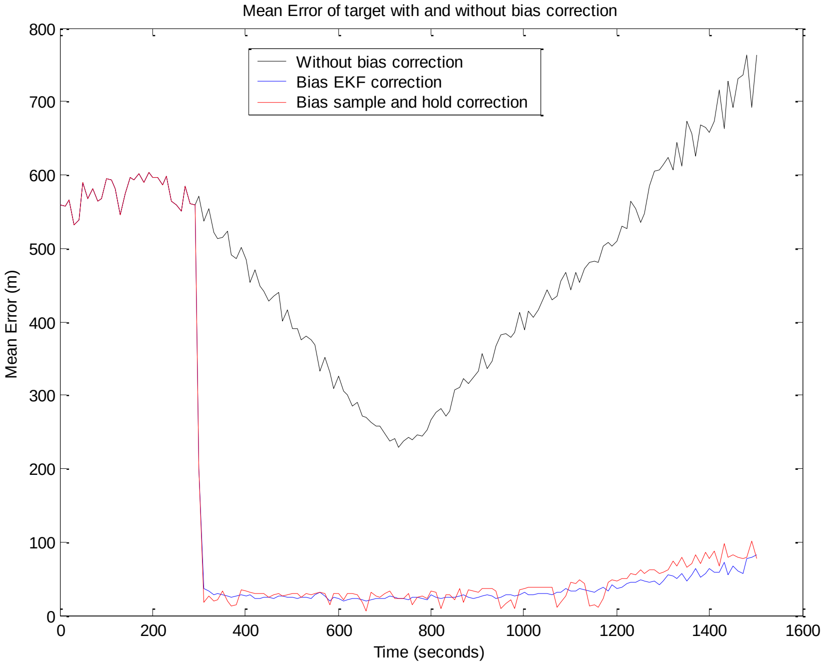


Figure 23. Case 2 - Mean error of target position for a constant bias with a sinusoidal component (β_0 = 0.2 *m*rads)

Table 9. Mean error in target position for constant bias with sinusoidal component (σ_s = 10 μrad, σ_p = 10 m, σ_β = 2 μrad, σ_ζ = 10 μrad)

Bias (β_0)	No bias correction	Simple bias correction	With EKF bias filter
None	32.5 m	35.8 m	32.9 m
.02 *m*rad	53.5 m	36.3 m	33.1 m
.2 *m*rad	445m	37.0 m	34.0m
2 *m*rad	4.44 km	46.7 m	49.4 m

5. TARGET TRACKING AND ESTIMATION

In this chapter we develop and compare three nonlinear algorithms to estimate the target state vector and its covariance. The first method uses an intercept calculation of the two target observation vectors to produce a three-dimensional position measurement of the target. This position provides a linear input measurement into a Linearized Kalman Filter (LKF) to produce a target state estimate. The second method uses the two dimensional nonlinear measurements of sensor azimuth and elevation as inputs to an Extended Kalman Filter (EKF). The third technique uses an Unscented Kalman Filter (UKF) for the target state estimate. Using the EKF and UKF we find individual state estimates from each sensor's data and then fuse them for a global estimate. These latter two methods allow the system to calculate the target location with only one satellite observation, whereas the LKF method requires that both satellites observe the target.

5.1. Target location by vector intercept

The first method of target location approximates the intercept point of the two three-dimensional line of sight vectors expressed in Cartesian coordinates as shown in Figure 24 [35]. In Figure 24 the x, y, and z axes represent the ECI reference frame. The target's position is denoted by $X = [X, Y, Z]'$, and the sensors, S_i, $i = 1,2$ are at position $S_i = [X^S, Y^S, Z^S]'$.

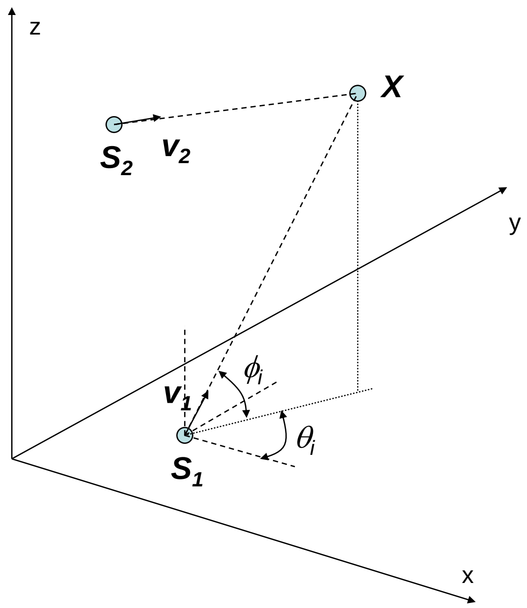


Figure 24. Target position from the intercept of two vectors

If we define

$$d_i = [d_{ix} \quad d_{iy} \quad d_{iz}]^T = [X - X_i^S \quad Y - Y_i^S \quad Z - Z_i^S]^T \tag{5-1}$$

as the distance between the target and sensor *i* then the line of sight is represented by a two-dimensional vector of azimuth, θ_i, and elevation, ϕ_i, such that

$$\begin{bmatrix} \theta_i \\ \phi_i \end{bmatrix} = \begin{bmatrix} \tan^{-1}\left({d_{iy}}/{d_{ix}} \right) \\ \tan^{-1}\left({d_{iz}}\Big/{\sqrt{d_{ix}^2 + d_{iy}^2}} \right) \end{bmatrix} . \tag{5-2}$$

This two-dimensional sensor line of sight measurement from the sensor to the target is converted to a three-dimensional unit vector through a polar to Cartesian conversion, or

$$v_i = \begin{bmatrix} v_{xi} \\ v_{yi} \\ v_{zi} \end{bmatrix} = \begin{bmatrix} \cos\theta_i \cos\phi_i \\ \sin\theta_i \cos\phi_i \\ \sin\phi_i \end{bmatrix} \tag{5-3}$$

Assuming we have two sensors tracking the same target from locations $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ one can find a point along each vector with the following equations.

$$X_1 = S_1 + av_1$$

$$X_2 = S_2 + av_2 \tag{5-4}$$

Where these vectors intercept we can set $\boldsymbol{X}_1 = \boldsymbol{X}_2$ and solve for a, which results in

$$a = \frac{|[S_2 - S_1] \times v_2|}{|v_1 \times v_2|}. \tag{5-5}$$

Then we can find the approximation of the target position as

$$X = S_1 + \frac{|[S_2 - S_1] \times v_2|}{|v_1 \times v_2|} \cdot v_1. \tag{5-6}$$

However, the line of sight vectors may not intercept due to the bias errors mentioned above. We found, however, that this process still provided a close approximation of target location proportional to the amount of the error and the range of the target. If more than two sensor satellites observe the target, a fused target location is calculated from sensor pairs with the individual pairs averaged.

5.2. Post correction target state estimation

Once bias is removed from the target line of sight a more accurate estimate of the target position can be obtained through the use of a nonlinear Kalman Filter. Here we develop three advanced methods of nonlinear filtering to estimate the target state. When measuring target position using the vector intercept method described above, we must use a Linearized Kalman Filter (LKF) to estimate target position. We can also obtain an independent estimate from each sensor using the EKF or UKF and then fuse these estimates into a single system estimate. These two methods of target estimation also allow the system to calculate the target location with only one satellite observation available, whereas the LKF method requires that both satellites observe the target. The use of each of these techniques is described in the following sections.

5.2.1. Space target dynamics

The state space model for a discrete-time stochastic system is of the general form

$$x_{k+1} = f[k, x_k, u_k, w_k]. \tag{5-7}$$

The motion of objects in orbit about the earth including ballistic missiles are modeled by the discrete-time linear dynamic equation

$$x_{k+1} = Ax_k + Bu_k + Gw_k \tag{5-8}$$

where x_k is the 6 dimensional state vector at time k represented as $[X_k\ \dot{X}_k\ Y_k\ \dot{Y}_k\ Z_k\ \dot{Z}_k]'$, A is the state transition matrix, u is a known input representing the gravitational effects acting on the target, and w is the process noise with covariance Q. The state transition matrices for a constant acceleration target are given as

$$\mathrm{A} = \begin{bmatrix} \begin{matrix} 1 & \Delta t \\ 0 & 1 \end{matrix} & 0 & 0 \\ 0 & \begin{matrix} 1 & \Delta t \\ 0 & 1 \end{matrix} & 0 \\ 0 & 0 & \begin{matrix} 1 & \Delta t \\ 0 & 1 \end{matrix} \end{bmatrix},$$

$$\mathrm{B} = \begin{bmatrix} \begin{matrix} {\Delta t^2}/{2} & \Delta t \\ 0 & 1 \end{matrix} & 0 & 0 \\ 0 & \begin{matrix} {\Delta t^2}/{2} & \Delta t \\ 0 & 1 \end{matrix} & 0 \\ 0 & 0 & \begin{matrix} {\Delta t^2}/{2} & \Delta t \\ 0 & 1 \end{matrix} \end{bmatrix},$$

and

$$G = \begin{bmatrix} \begin{matrix} {\Delta t^2}/{2} \\ \Delta t \end{matrix} & 0 & 0 \\ 0 & \begin{matrix} {\Delta t^2}/{2} \\ \Delta t \end{matrix} & 0 \\ 0 & 0 & \begin{matrix} {\Delta t^2}/{2} \\ \Delta t \end{matrix} \end{bmatrix}, \tag{5-9}$$

where Δt is the sampling interval. The gravity term is given by $u = g/alt$, where $g = 9.8$ m/s^2, and alt is the altitude of the target and provides the scaling factor for the gravity term. The process noise, w, accounts for the inaccurate modeling of the true system dynamics and is added to the state to model the missile accelerations due to maneuvers with a covariance matrix Q,

$$Q = \mathrm{diag}[\ (\sigma_X)^2 \quad (\sigma_Y)^2 \quad (\sigma_Z)^2]. \tag{5-10}$$

5.2.2. Infrared sensor measurement model

The general observation equation of the target's position corrupted by additive noise is

$$z_k = h[k, x_k, v_k], \tag{5-11}$$

or

$$z_k = h(x_k) + v_k \tag{5-12}$$

where $h(x_k)$ is the observation matrix and v_k is the observation noise with variance R consisting of azimuth and elevation error variance, $\sigma^2{}_\theta$ and $\sigma^2{}_\phi$. In our model, the assumption is made that the azimuth and elevation of the sensor drive motors align with the *y*,*z* grid of the sensor array. If not, an additional rotational conversion is required.

5.2.3. Linearized Kalman Filter

We will apply the LKF to estimate the target state when using the observation vector intercept method of target location discussed in section 5.1. The measurement associated with the intercept position is a three-dimensional coordinates in the ECI reference frame. To properly express this observation in the linearized case the measurement matrix and error covariance must be transformed into a Cartesian coordinate frame. We begin this transformation by expressing the estimate of the state vector in this LKF as a derivation from the nonlinear Kalman Filter such that

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K\left(z_k - h\left(\hat{x}_{k|k-1}\right)\right)$$

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K\left(h\left(\hat{x}_k + v_k - h\left(\hat{x}_{k|k-1}\right)\right)\right) \tag{5-13}$$

or

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_1\left(h^{-1}(z_k) - \hat{x}_{k|k-1}\right) \tag{5-14}$$

where

$$K_1 = P_{k|k-1}H_1S_1^{-1},\ S_1 = H_1P_{k|k-1}H_1^{-1} + R_1$$

$$H_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \tag{5-15}$$

$h^{-1}(z_k) = [r\cos\theta\cos\phi \quad r\sin\theta\cos\phi \quad r\sin\phi]$, and R_1 is the transformed measurement error covariance

$$R_1 = \begin{bmatrix} \frac{\partial x}{\partial r} & \frac{\partial x}{\partial \theta} & \frac{\partial x}{\partial \phi} \\ \frac{\partial y}{\partial r} & \frac{\partial y}{\partial \theta} & \frac{\partial y}{\partial \phi} \\ \frac{\partial z}{\partial r} & \frac{\partial z}{\partial \theta} & \frac{\partial z}{\partial \phi} \end{bmatrix} \begin{bmatrix} \sigma_r^2 & 0 & 0 \\ 0 & \sigma_\theta^2 & 0 \\ 0 & 0 & \sigma_\phi^2 \end{bmatrix} \begin{bmatrix} \frac{\partial x}{\partial r} & \frac{\partial x}{\partial \theta} & \frac{\partial x}{\partial \phi} \\ \frac{\partial y}{\partial r} & \frac{\partial y}{\partial \theta} & \frac{\partial y}{\partial \phi} \\ \frac{\partial z}{\partial r} & \frac{\partial z}{\partial \theta} & \frac{\partial z}{\partial \phi} \end{bmatrix}^T \tag{5-16}$$

with

$$\begin{bmatrix} \frac{\partial x}{\partial r} & \frac{\partial x}{\partial \theta} & \frac{\partial x}{\partial \phi} \\ \frac{\partial y}{\partial r} & \frac{\partial y}{\partial \theta} & \frac{\partial y}{\partial \phi} \\ \frac{\partial z}{\partial r} & \frac{\partial z}{\partial \theta} & \frac{\partial z}{\partial \phi} \end{bmatrix} = \begin{bmatrix} \cos\theta\cos\phi & -r\sin\theta\cos\phi & -r\sin\theta\cos\phi \\ \sin\theta\cos\phi & r\cos\theta\cos\phi & -r\sin\theta\sin\phi \\ \cos\phi & 0 & r\cos\phi \end{bmatrix}. \tag{5-17}$$

The transformation of the error covariance from R to R_1 requires an estimate of the range error variance σ_r^2. However, the bearing only tracking system does not measure range directly. Therefore, a value must be determined either through an estimate of the accuracy of intercept point calculation, or directly from the two bearing measurements.

Figure 25 shows a two dimensional representation of the bearing measurements of the IR sensors where the error in the angle of the measurement is given by the angle standard deviation σ_θ or σ_ϕ. The parallelogram like shape resulting from the overlap of the observation errors can be approximated by an ellipse, as shown. The minor axis of this ellipse remains relatively constant and is represented by the bearing accuracy of the referenced sensor. The bearing accuracy of the cross-LOS sensor determines the major-axis length and is proportional to the angle α, between the sensor lines of sight. When the $\alpha = 90$ degrees, the ellipse becomes near circular depending upon the range to the individual sensors [35].

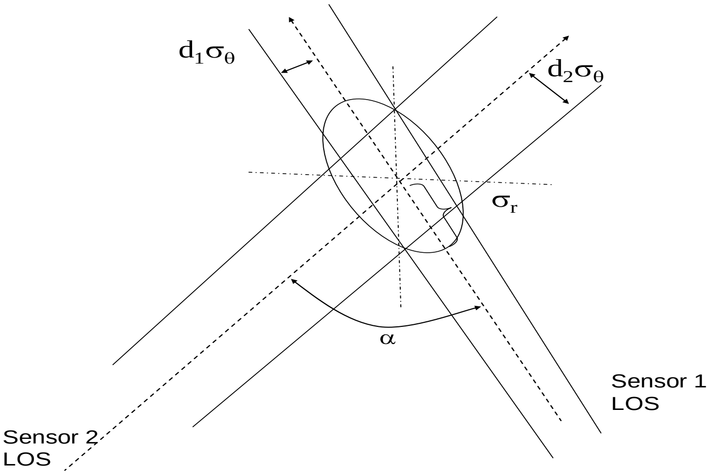


Figure 25. Line of sight intercept geometry.

It can be shown that the length of the semi-major axis σ_r can be approximated by:

$$\sigma_r = {d_2\sigma_\theta}/{\cos\left({\alpha}/{2}\right)}, \qquad 0 < \alpha \leq 90 \tag{5-18}$$

As α goes to zero the ellipse becomes infinitely long. Thus the satellites must have a baseline separation with the optimal at 90 degrees to give the smallest ellipse volume. The true orientation and size of the ellipse represents the cross-correlation of the sensor measurement errors but this method provides a close approximation. For the 3 dimensional case α is given by

$$\alpha = \cos^{-1}(\nu_1 \cdot \nu_2). \tag{5-19}$$

Figure 26 plots the value for the range variance over time for the test algorithm. As expected the magnitude is proportional to the range of the sensor to the target due to the $d_i\sigma_\theta$ term.

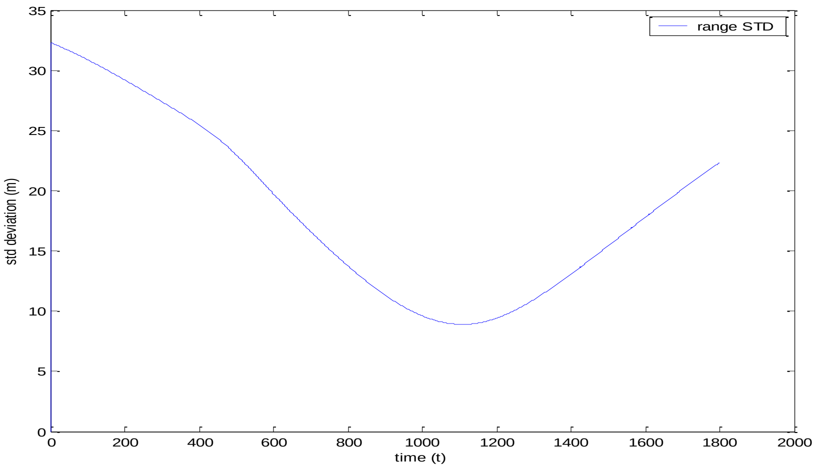


Figure 26. Range standard deviation (σ_r) over time

5.2.4. Extended Kalman Filter (EKF)

As stated above, the state space model for a discrete-time stochastic system is of the general form

$$x_{k+1} = f[k, x_k, u_k, v_k], \tag{5-20}$$

with a measurement model

$$z_k = h[k, x_k, w_k], \tag{5-21}$$

where *u(k)* is a known input, *v(k)* is process noise with covariance *Q(k),* and *w(k)* is the measurement noise with covariance *R(k).* In a nonlinear system at least one of the two functions in nonlinear. As with the linear Kalman Filter, all noise sequences are assumed to be white and independent with respect to each other and the initial state has a known probability distribution function that is also independent of the noise.

A first-order Taylor series expansion of (5-18) provides a linear representation of the nonlinear system such that

$$x_{k+1} = f\left[k, \hat{x}_{(k|k)}\right] + f(x_k)\left[x_k - \hat{x}_{(k|k)}\right] + HOT + v_k, \tag{5-22}$$

where

$$f(x_k) \stackrel{\text{def}}{=} [\nabla_{\mathrm{x}} f(x,k)']'|_{x=\hat{x}_{(k|k)}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix} \stackrel{\text{def}}{=} F_k, \tag{5-23}$$

and HOT are the higher order terms. The predicted state is

$$\hat{x}_{(k+1|k)} = f[k, \hat{x}_{(k|k)}], \tag{5-24}$$

and predicted state covariance is

$$P_{(k|k+1)} = f(x_k) P_{k|k} f(x_k)' + Q_k. \tag{5-25}$$

The measurement prediction follows a similar expansion of (5-22) to

$$z_{k+1} = h\left[k+1, \hat{x}_{(k+1|k)}\right] + h(x_{k+1})\left[x_{k+1} - \hat{x}_{(k+1|k)}\right] + HOT + w_k \tag{5-26}$$

where

$$h(x_{k+1}) \stackrel{\text{def}}{=} [\nabla_{\mathrm{x}} h(x,k+1)']'|_{x=\hat{x}_{(k+1|k)}} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \cdots & \frac{\partial h_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial h_m}{\partial x_1} & \cdots & \frac{\partial h_m}{\partial x_n} \end{bmatrix} \stackrel{\text{def}}{=} H_{k+1}. \tag{5-27}$$

The predicted measurement is

$$\hat{z}_{(k+1|k)} = h[k+1, \hat{x}_{(k+1|k)}], \tag{5-28}$$

and predicted measurement covariance is

$$S_{k+1} = f(x_{k+1})P_{k+1|k}f(x_{k+1})' + R_{k+1}. \qquad (5\text{-}29)$$

Some difficulties in using the EKF is that the nonlinear transformations may introduce bias and the predicted state may not be accurate since only the first order terms are used in the prediction and measurements.

5.2.5. The Unscented Kalman Filter (UKF)

The UKF uses a small set of carefully chosen sample points to represent the Gaussian state distribution. The sample points are then propagated forward in time using the actual nonlinear transformation matrix. The mean and the covariance of these transformed points provide an estimate with the accuracy of the 3rd order Taylor series expansion. The sample points are selected by using a Gaussian distribution with a weighted sample mean and covariance [11].

To calculate the estimation of the state vector $\boldsymbol{x}(k+1)$ using the UKF one begins by developing a matrix χ of $2L+1$ sigma vectors χ_ι , each with a weight W_i, where L is the dimension of the state vector **x**. The sigma vectors and their weights are calculated by

$$\chi_0 = \bar{x}$$

$$\chi_i = \bar{x} - \left(\sqrt{(L+\lambda)P_x}\right)_i \qquad i=1\ldots,L \qquad (5\text{-}30)$$

$$\chi_i = \bar{x} - \left(\sqrt{(L+\lambda)P_x}\right)_i \qquad i=L+1,\ \ldots,\ 2L$$

$$W_0^{(m)} = \lambda/(L+\lambda)$$

$$W_0^{(c)} = \lambda/(L+\lambda) + (1-\alpha^2+\beta) \qquad (5\text{-}31)$$

$$W_{0i}^{(m)} = W_i^c = 1/[2(L+\lambda)]_{,} \qquad i = 1,\ldots,2L\ ,$$

where $(L + \lambda)P_x$ is the ith row or column of the matrix square root of $(n_x + k)P_x$. The weights are normalized.

Each sigma point is propagated through the nonlinear function

$$x_i\,(k+1) = f(x_i(k)), \qquad i = 0, \ldots, 2L. \tag{5-32}$$

The estimated mean and covariance of x are computed as the sum of the propagated points

$$\hat{x} = \sum_{i=0}^{2L} W_i \chi_i \tag{5-33}$$

$$P_x = \sum_{i=0}^{2L} W_i (\chi_i - \hat{x})\,(\chi_i - \hat{x})^T. \tag{5-34}$$

Implementation of the above algorithm to find the next estimate at $\hat{x}(k+1)$ with the current estimate and its covariance $\hat{x}(k)$ and $P_x(k|k)$follows as

1) Compute sigma points χ_i and their weights W_i , (i= 0,…2L) using (5-30) and (5-31) with α=1 , β=2, and λ=0, using $\hat{x}(k)$ and $P_x(k|k)$. The matrix square root in (5-30) is found using Cholesky factorization.
2) The sigma points are propagated using the state equation such that

$$\chi_i(k+1|k) = f[\chi_i(k|k)] + Q(k) \tag{5-35}$$

3) From these new points, compute the predicted state and covariance

$$\hat{x}(k+1|k) = \sum_{i=0}^{2L} W_i \chi_i\,(k+1|k) \tag{5-36}$$

$$P(k+1|k) = \sum_{i=0}^{2L} W_i[(\chi_i(k+1) - \hat{x}(k+1|k)] \cdot [(\chi_i(k+1) - \hat{x}(k+|k)]^T + Q(k) \tag{5-37}$$

4) Next predict the observation sigma points

$$\zeta_i(k+1|k) = h[\zeta_i(k|k)], \tag{5-38}$$

5) and the observation estimate and covariance follows

$$\hat{y}(k+1|k) = \sum_{i=0}^{2L} W_i \zeta_i\,(k+1|k) \tag{5-39}$$

$$P_{yy}(k+1|k) = \sum_{i=0}^{2L} W_i[(\zeta_i(k+1) - \hat{y}(k+1|k)] \cdot [(\zeta_i(k+1) - \hat{y}(k+|k)]^T + R(k+1) \tag{5-40}$$

$$P_{xy}(k+1|k) = \sum_{i=0}^{2L} W_i[(\chi_i(k+1) - \hat{x}(k+1|k)] \cdot [(\zeta_i(k+1) - \hat{y}(k+|k)]^T \tag{5-41}$$

6) Compute the UKF gain and updated state and state covariance similar to the standard Kalman Filter

$$K(k+1) = P_{xy}P_{yy}^{-1}$$

$$\hat{x}(k+1|k+1) = \hat{x}(k+1|k) + K(k+1)[y(k+1) - \hat{y}(k+1|k)] \tag{5-42}$$

$$P(k+1|k+1) = P(k+1|k) - K(k+1)P_{yy}K^T(k+1)$$

As mentioned above, the primary difference in using the EKF and UKF filters is in the observation matrix *h*, which is described by equation (5-2), or

$$h_i(x) = \begin{bmatrix} \theta_i \\ \phi_i \end{bmatrix} = \begin{bmatrix} \tan^{-1}\left({d_{iy}}\big/{d_{ix}} \right) \\ \tan^{-1}\left({d_{iz}}\Big/{\sqrt{{d_{ix}}^2 + {d_{iy}}^2}} \right) \end{bmatrix}, \tag{5-43}$$

where *i* is the respective sensor. Additionally, the sensor error covariance matrix, *R_i* becomes

$$R_i = \begin{bmatrix} \sigma_{\theta_i}^2 & 0 \\ 0 & \sigma_{\phi_i}^2 \end{bmatrix}. \tag{5-44}$$

5.2.6. Sensor Fusion

Following the use of the EKF, and UKF, if more than one satellite observation is available, the target state estimate is improved by fusing the observations as described in [12]. Figure 27 shows the general update and estimation cycle for a two sensor fusion system without feedback. Integration of other sensor, such as radars would follow the same techniques. Other sensor fusion options include feedback and cyclical fusion that are not explored in this study.

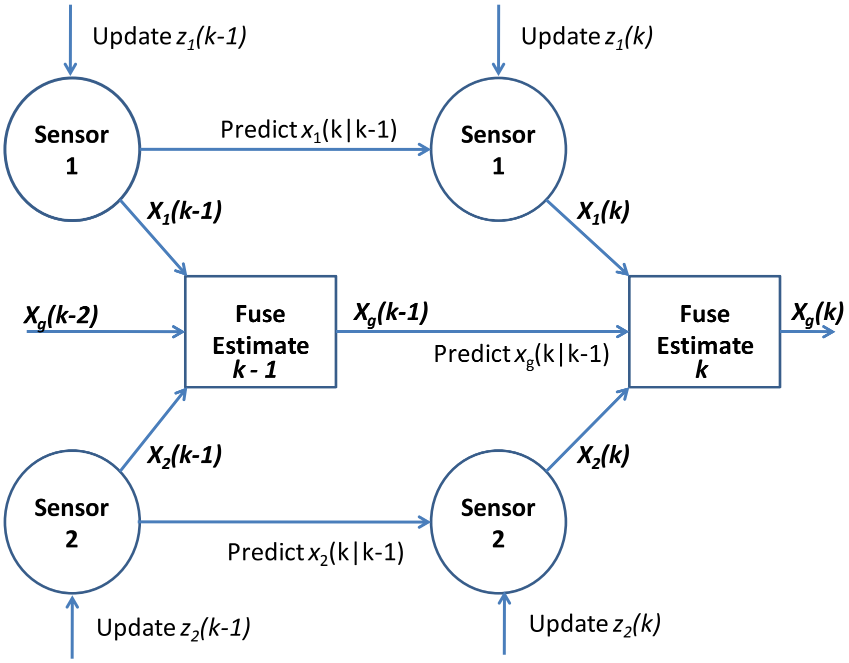


Figure 27. Two sensor fusion without feedback example.

The general fusion equations for two sensors without feedback are

$$P_{g_{k|k-1}} = \mathrm{A} P_{g_{k-1|k-1}} \mathrm{A}^T + Q \tag{5-45}$$

$$\hat{x}_{g_{k|k-1}} = \mathrm{A} \hat{x}_{g_{k-1|k-1}} \tag{5-46}$$

$$P_{g_{k|k}} = \left[P_{1}^{-1}{}_{k|k} + P_{2}^{-1}{}_{k|k} - P_{g}^{-1}{}_{k|k-1}\right]^{-1} \tag{5-47}$$

$$\hat{x}_{g_{k|k}} = P_{g_{k|k}} P_{g}^{-1}{}_{k|k-1} \hat{x}_{g_{k|k-1}} + P_{g_{k|k}} \left[P_{1}^{-1}{}_{k|k} \hat{x}_{1_{k|k}} + P_{2}^{-1}{}_{k|k} \hat{x}_{2_{k|k}}\right]. \tag{5-48}$$

5.3. Post-correction target state estimator results

Following the correction of the target LOS described in Chapter 4, the resulting target observations were used as measurement inputs to the three nonlinear filters discussed above. We adapted the MATLAB Recursive Bayesian Estimation Laboratory (ReBEL) Toolkit© [45] developed by R. van der Merwe for performing the simulations used in this study. The underlying filter subroutines for the EKF and UKF were unaltered from the baseline program. For the EKF, in place of finding the Jacobian matrices for the state and measurement functions, the ReBEL program used in this study approximates linearization of the state function, by propagating the state and measurements in 10^6 steps per time interval. The results, mean error (top value) and standard deviation are shown below in Tables 10 and 11 for the two bias cases respectively.

Table 10. Mean error in target position for pure sinusoidal bias (σ_s = 10 μrad, σ_p = 10 m, σ_β = 2 μrad, σ_ζ = 10 μrad)

Bias (β_0)	No bias correction	With EKF bias filter and LKF	With EKF bias filter and EKF	With EKF bias filter and UKF
None	40.6 m	32.0 m	25.6 m	25.6 m
.02 *m*rad	52.9 m	32.0 m	25.8 m	25.4 m
.2 *m*rad	489 m	33.5 m	27.1 m	26.9 m
2 *m*rad	4.88 km	49.4 m	44.1 m	43.9 m

Table 11. Mean error in target position for constant bias with sinusoidal component ($\sigma_s = 10\ \mu$rad, $\sigma_p = 10$ m, $\sigma_\beta = 2\ \mu$rad, $\sigma_\zeta = 10\ \mu$rad)

Bias (β_0)	No bias correction	With EKF bias filter and LKF	With EKF bias filter and EKF	With EKF bias filter and UKF
None	32.5 m	26.5 m	21.8 m	21.8 m
.02 *m*rad	53.3 m	26.6 m	22.0 m	21.8 m
.2 *m*rad	445.0 m	27.8 m	23.3 m	23.2 m
2 *m*rad	4.44 km	46.1 m	44.3 m	44.1 m

The little difference between the filters is evident in Figure 28. The individual sensor state estimates from the EKF and UKF estimators were fused to provide a common state estimate as described above.

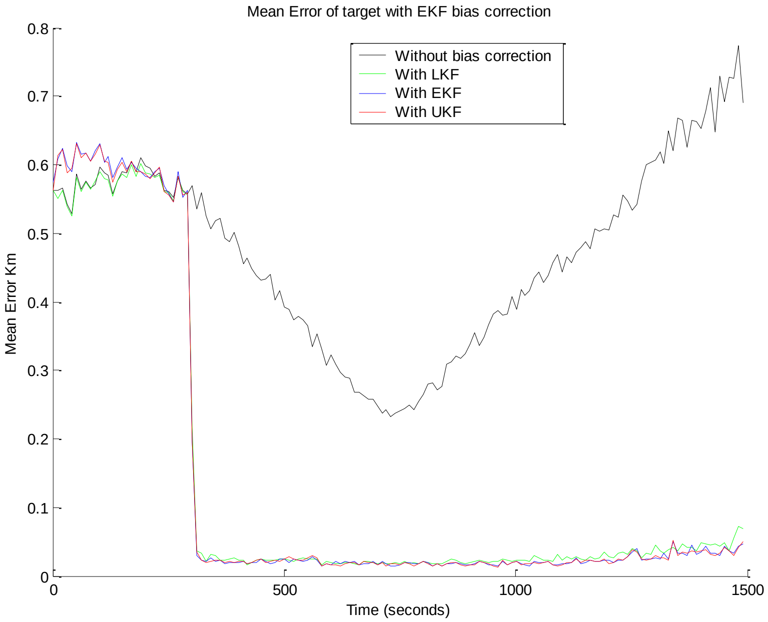


Figure 28. Comparison of mean error of target position using LKF, EKF, and UKF estimation after EKF filter bias correction.

To understand the impact of the bias removal process, one question to consider is whether the nonlinear estimators can provide adequate target position estimates without bias correction. If an estimator can provide enough improvement and bring the target error within prescribed amounts, we may not need to perform bias correction after all. Therefore, we performed nonlinear estimation on the sensor observations using the LKF, EKF and UKF prior to bias correction. As can be seen in Figures 29-31, the nonlinear filters do not provide much improvement. We can see how in Figure 29 the LKF provides very little improvement and is consistent in what we found in section 4.2. Additionally, the EKF (Figure 30) and UKF (Figure 31) do provide a bit of improvement during a short period of the missile flight, but the target position error is still significant compared to the corrected values. The fact that there is not much difference between the three filters is somewhat surprising. This similarity between filters might be caused by the chosen scenario (i.e., nonlinearity, sampling interval, sensor accuracy, etc). In our case the scenario is very well behaved. For instance, the sampling interval is small (1 second), this means that there is not much movement of the target or the corresponding observations between time steps to make the problem highly nonlinear. Also, the sensor error is relatively small compared to the bias error. This allows for very small target position errors when the bias error is removed.

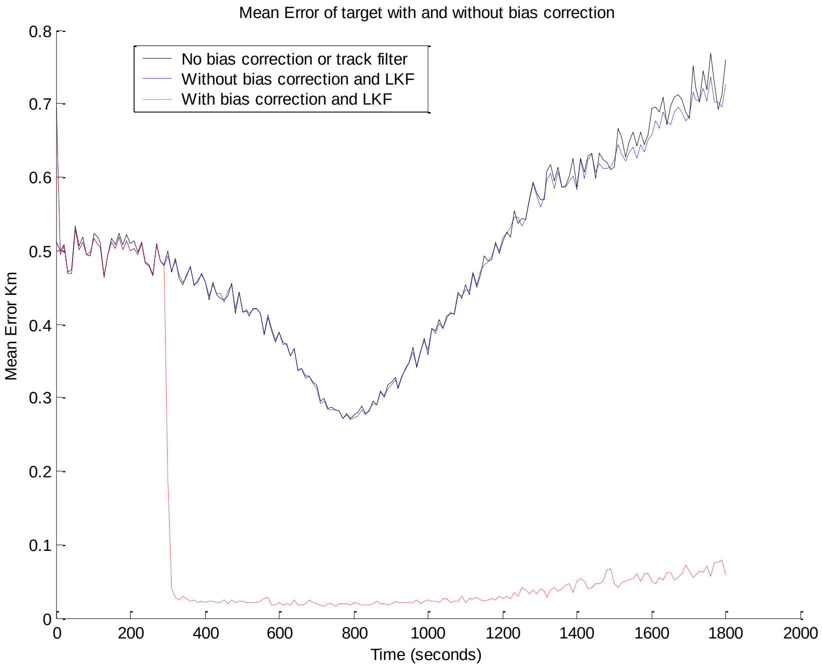


Figure 29. Mean error of target position with and without bias correction using LKF filter

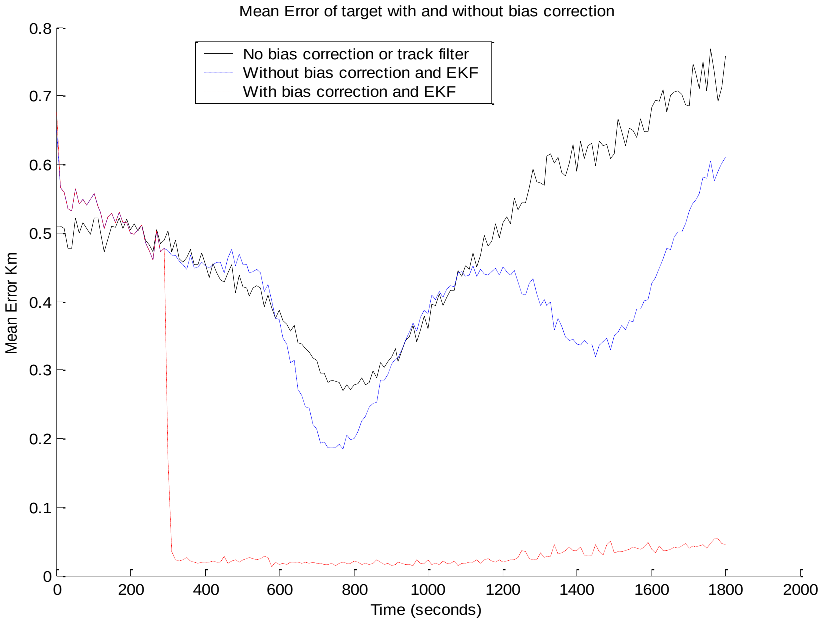


Figure 30. Mean error of target position with and without bias correction using EKF filter

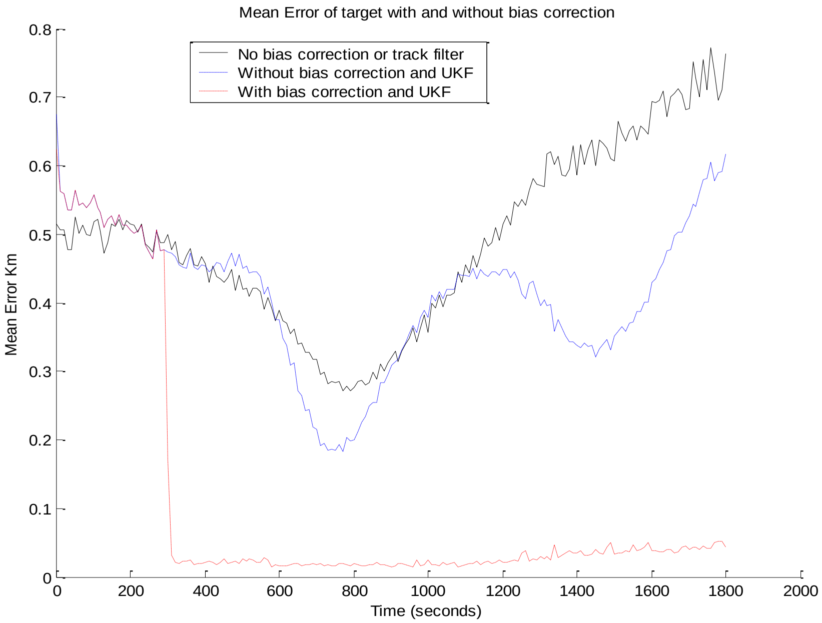


Figure 31. Mean error of target position with and without bias correction using UKF filter

Table 12. Mean error in target position for 50 μrad sensor error (pure sinusoidal bias).
(σ_s = 50 μrad, σ_p = 10 m, σ_β = 2 μrad, σ_ζ = 10 μrad)

Bias (β_0)	LKF	EKF	UKF
None	103 m	83.7 m	81.3 m
.02 *m*rad	104 m	82.9 m	81.1 m
.2 *m*rad	104m	85.0 m	83.3 m
2 *m*rad	117m	101 m	100 m

We also varied the sensor errors and sampling intervals in order to understand their impacts on the estimation performance. For instance, when sensor error ($\sigma_{\theta,\phi}$), or the time step between observations are increased we begin to see separation in the performance of the nonlinear filters. Figures 32 and 33 show the mean error in target position when sensor error is 50 μrad and 100 μrad respectively. In the case of a 100 μrad sensor error, the bias estimate is greatly affected by the sensor error and this poorer

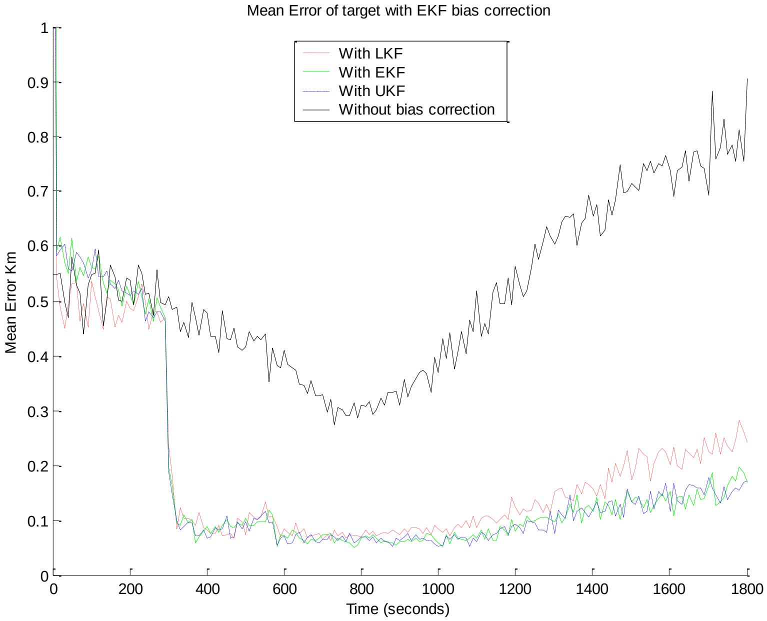


Figure 32. Comparison of mean error of target position using LKF, EKF, and UKF estimation after EKF filter bias correction for 50 μrad sensor error.

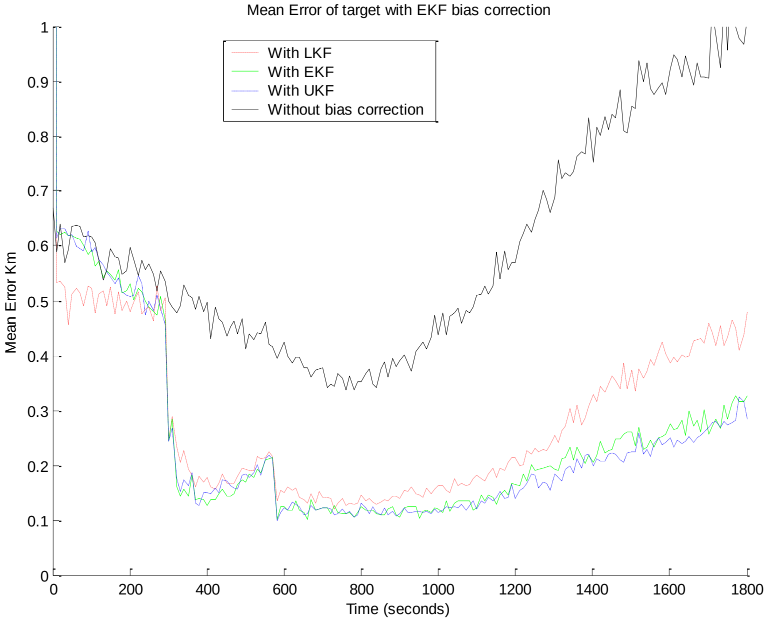


Figure 33. Comparison of mean error of target position using LKF, EKF, and UKF estimation after EKF filter bias correction for 100 μrad sensor error.

estimate contributes greatly to the position error. These results can be seen in the diagrams of bias estimation shown in Appendix A, specifically figures A-11 and A-12.

We also can see in Figures 34 and 35 the significant change in performance of the LKF when the observation time step is increased to 5 seconds and 10 seconds. Table 13 shows the mean error for the four different parameters as compared to the standard development case for sensor bias (β_0) of .2 *m*rad. Essentially, we found the resulting performance somewhat expected and they seem to indicate that LKF is not as robust as others. However, even with the poorer performance, from Table 14 we can see that the nonlinear UKF filter still provide an improved condition over the non-filtered solution, especially with increased senor error (A more complete table of results can be found in Tables A-3 and A-4 of Appendix A). In Table 14 we can also see how the senor error has a greater effect on target position estimation than the time step difference. This is also expected whereas the time step mostly impacts relative nonlinearity of the system and the number of detected stars.

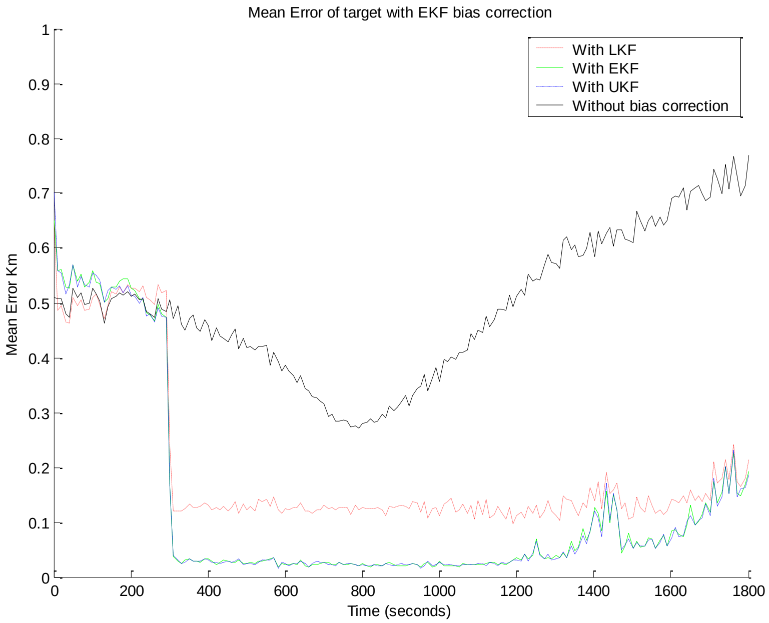


Figure 34. Comparison of mean error of target position using LKF, EKF, and UKF estimation after EKF filter bias correction for 5 second measurement time step.

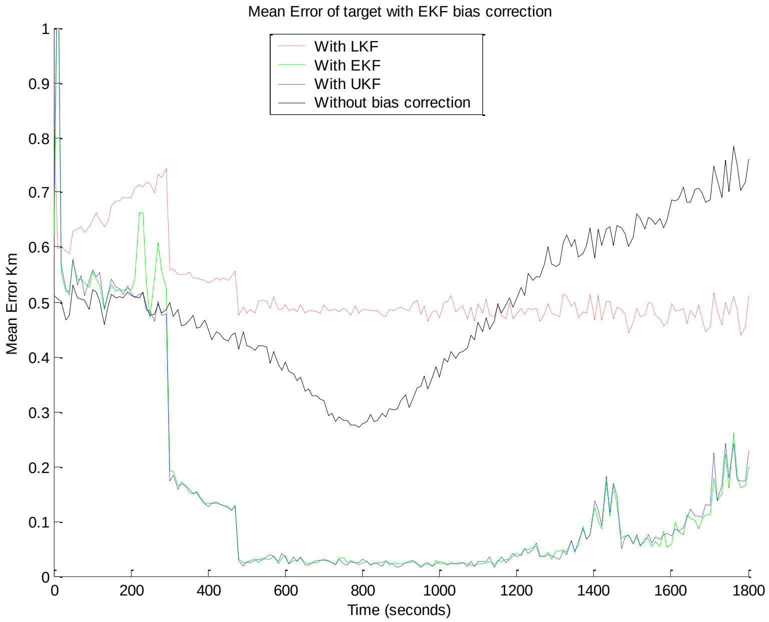


Figure 35. Comparison of mean error of target position using LKF, EKF, and UKF estimation after EKF filter bias correction for 10 second measurement time step.

Table 13. Comparison of mean error in target position for various system variables for common bias (β_0) = 0.2 *m*rad (σ_p = 10 m, σ_β = 2 μrad, σ_ζ = 10 μrad).

	σ_s = 10 μrad T_s = 1 sec	σ_s = 50 μrad T_s = 1 sec	σ_s = 100 μrad T_s = 1 sec	T_s = 5 sec σ_s = 10 μrad	T_s = 10 sec σ_s = 10 μrad
LKF	33.9 m	104m	237 m	185 m	519 m
EKF	27.1 m	85.0 m	180 m	113 m	139 m
UKF	26.9 m	83.3 m	172 m	114 m	137 m

Table 14. Comparison of mean error in target position for non-filtered and filtered estimates.

Bias	σ_s = 50 μrad T_s = 1 sec		σ_s = 100 μrad T_s = 1 sec		T_s = 5 sec σ_s = 10 μrad		T_s = 10 sec σ_s = 10 μrad	
	No filter	UKF	No filter	UKF	No filter	UKF	No filter	UKF
None	148 m	81.3 m	368 m	171 m	41 m	35.2 m	42.1 m	37.1 m
.02 *m*rad	148 m	81.1 m	368 m	169 m	43.6 m	38.1 m	46.1 m	41.2 m
.2 *m*rad	149 m	83.3 m	368 m	171 m	118 m	114 m	140 m	137 m
2 *m*rad	156 m	100 m	377 m	197 m	959 m	958 m	1.22 km	1.22 km

6. FILTER PERFORMANCE MEASURES

In this chapter we analyze how well the nonlinear filters perform against two performance measures, the Posterior Cramer-Rao Lower Bound (PCLRB) and a filter consistency metric. We then test the correction and estimation algorithm against a number of different scenarios representing various missile trajectories and sensor satellite orbits using a quantitative MATLAB simulation.

6.1. Posterior Cramer-Rao lower bound (PCLRB) performance

The PCRLB provides a mean square error bound on the performance of an unbiased estimator. For a tracking estimator, the PCRLB enables one to determine a lower bound on the optimal performance on the achievable accuracy of the target state estimate. Let C_k be the error covariance of an unknown and random target state x_k, which is estimated by an unbiased estimator $\hat{x}_k$. The PCRLB is defined as the inverse of the Fisher Information matrix J_k, or

$$C_k \triangleq \mathbb{E}[(x_k - \hat{x}_k)(x_k - \hat{x}_k)^T] \geq J_k^{-1}, \tag{6-1}$$

where J_k as defined as a n x n matrix with the elements [46].

$$J_{ij} = \mathbb{E}\left[-\frac{\partial^2 \log P_{x,\hat{x}}(x,\hat{x})}{\partial \hat{x}_i \partial \hat{x}_j}\right] \qquad i,j = 1, \ldots, n. \tag{6-2}$$

The bound is referred to as the 'posterior CRLB' due to the non-zero process noise in the state dynamics. The Fisher information matrix can be shown to represent a system with linear target dynamics at time k+1 by the following recursion [47]:

$$J_{k+1} = D_k^{22} - D_k^{21}(J_k + D_k^{11})^{-1} D_k^{12},\ (k>0) \tag{6-3}$$

where, for the additive Gaussian form of the process and measurement noise have the following components:

$$\begin{aligned} D_k^{11} &= \mathbb{E}\{F_k^T Q_k^{-1} F_k\} \\ D_k^{12} &= \mathbb{E}\{F_k^T\} Q_k^{-1} = [D_k^{21}]^T \\ D_k^{22} &= Q_k^{-1} + \mathbb{E}\{H_{k+1}^T R_{k+1}^{-1} H_{k+1}\}. \end{aligned} \tag{6-4}$$

where

$$F_k = \left[\nabla_{x_k} f_k^T(x_k)\right]^T \tag{6-5}$$

$$H_k = \left[\nabla_{x_k} h_k^T(x_k)\right]^T \tag{6-6}$$

are Jacobians of the nonlinear functions $f_k(x_k)$ and $h_k(x_k)$ respectively, evaluated at the true value of x_k. The expectations above are with respect to x_k. Since our target dynamics are nearly linear we can assume that $f_k(x_k)=A_k$, and (6-3) is simplified to [48]:

$$J_{k+1} = (\Sigma_k + A_k J_k^{-1} A_k^T)^{-1} + J_Z(k+1). \tag{6-7}$$

Where $J_z(k+1)$ is the measurement contribution, A_k is the state process transformation matrix and $\Sigma_k = GQG^T$ is the process noise covariance. For more than one sensor contribute to the ultimate state estimate, we must consider the contribution of all these sensors. Thus, if we have M sensors providing measurements at time k, this contribution is formulated as

$$J_z(k) = \mathbb{E}_M[J_z(k: M)] = \sum_M p(M) J_z(k: M) \tag{6-8}$$

where *p(M)* is the probability mass function of *M* and *J_z(k:M)* is the measurement contribution [48]. *Jz(k:M)* is found from the Jacobian of the observation matrix $H_k = \nabla_{x_k} h(x_k)$, where $\nabla_{x_k} h(x_k)$ represents the matrix of the first partial derivatives of the vector h_k with respect to the state vector X_k at time *k*, such that [7]:

$$J_z(k: M) = H_{M_k}^T R_{M_k}^{-1} H_{M_k}. \tag{6-9}$$

In our case, we have two identical sensors providing measurements at each time step, and therefore *p(M)*=.5, and from (6-8) and (6-9) we get

$$J_z(k) = \sum_{l=1}^{2} .5 * H_{l_k}^T R^{-1} H_{l_k}, \tag{6-10}$$

where $h(x_k)$ is given by (5-43) and *R* by (5-44). The Jacobian of $h(x_k)$, at each time step, *k*, given by

$$[H_k]_{ij} = \frac{\partial h_i(x_k)}{\partial x_k(j)} \tag{6-11}$$

has components

$$[H_k]_{11} = \frac{-(Y_k - Y_k^S)}{[(X_k - X_k^S)^2 + (Y_k - Y_k^S)^2]}$$

$$[H_k]_{13} = \frac{(X_k - X_k^S)}{[(X_k - X_k^S)^2 + (Y_k - Y_k^S)^2]}$$

$$[H_k]_{12} = [H_k]_{14} = [H_k]_{15} = [H_k]_{16} = 0$$

$$[H_k]_{21} = \frac{-(X_k - X_k^S)(Z_k - Z_k^S)}{\rho_k^2 [(X_k - X_k^S)^2 + (Y_k - Y_k^S)^2]^{1/2}}$$

$$[H_k]_{23} = \frac{-(Y_k - Y_k^S)(Z_k - Z_k^S)}{\rho_k^2 [(X_k - X_k^S)^2 + (Y_k - Y_k^S)^2]^{1/2}}$$

$$[H_k]_{25} = \frac{[(X_k - X_k^S)^2 + (Y_k - Y_k^S)^2]^{1/2}}{\rho_k^2}$$

$$[H_k]_{22} = [H_k]_{24} = [H_k]_{26} = 0. \tag{6-12}$$

In the above equations, ρ_k is the distance between the target and the sensor at the sampling time k, X, Y, Z are the target position components, and X^S, Y^S, and Z^S are the sensor position components.

When we compare the resulting PCRLB with the error variance in Figures 32 and 33 we can see how the variance closely follows the lower bound but does not quite match the bound. This discrepancy is most likely due to non-Gaussian errors introduced as a result of coordinate frame transformations and bias correction. In figure 33 we can see the transition at t=300 seconds when the system begins observing stars and correcting for observation bias. This step results from the celestial observations helping correct for the sensor Gaussian observation error in addition to the bias error.

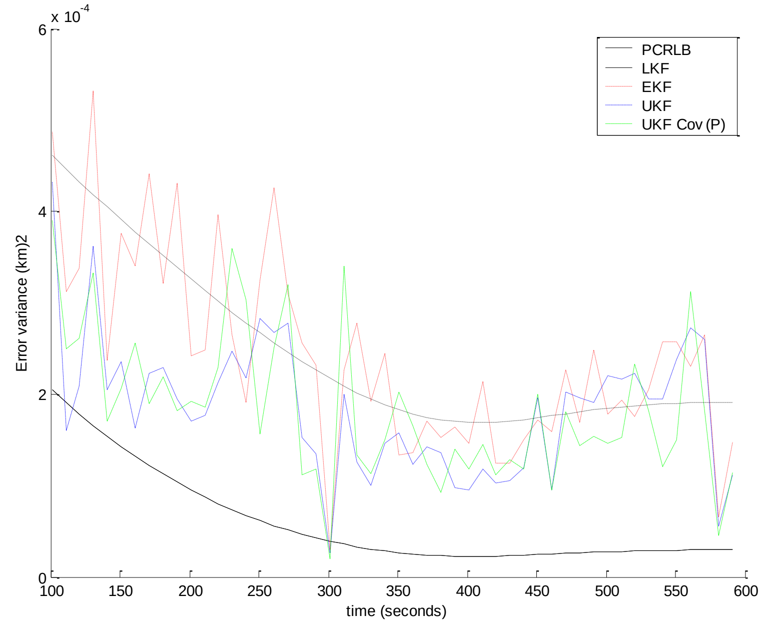


Figure 36. Error variance vs PCRLB for no bias case.

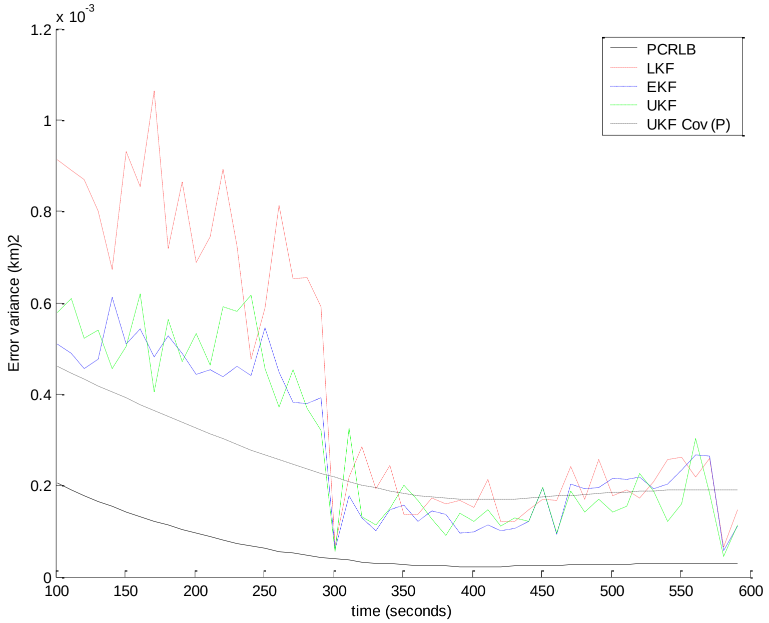


Figure 37. Error variance vs PCRLB for .2 *m*rads bias error.

6.2. Filter Consistency

Another measure of filter consistency involves weighting the estimate error by the inverse of the covariance matrix such that the normalized (state) estimation error squared (NEES) [8] can be obtained by:

$$\epsilon(k) = \left(x(k) - \hat{x}(k|k)\right)^T P(k|k)^{-1}\left(x(k) - \hat{x}(k\,|\,k)\right), \tag{6-13}$$

With a sufficiently large, number of Monte Carlo runs, N, the value of $\epsilon(k)$ converges to a mean, $\bar{\epsilon}(k)$. Therefore, a consistent filter will meet the following condition:

$$\mathbb{E}[\epsilon(k)] \cong \bar{\epsilon}(k) = \frac{1}{N}\sum_{i=1}^{N}\epsilon_i(k) \cong n_x, \tag{6-14}$$

where n_x is the dimension of the state, x.

For our study $n_x = 6$. Figures 34 to 36 show the values of epsilon for N=200 as the three estimators refine the target state vector. The results for the EKF and UKF estimators are consistent with the expected result. Specifically $\bar{\epsilon} = 6.9$ for the UKF and $\bar{\epsilon} = 6.8$ for the EKF. However, we can see in Figure 33 how the LKF estimator is well off the expected NEES value. The reason for this result is that the velocity components of the covariance matrix, P, are ill conditioned due to the Polar to Cartesian conversion of the measurements in order to linearlize the filter. We also notice that there are spikes in the NEES values at $t = 300$ seconds in the EKF and UKF cases. This is spike is the result of beginning error correction using star observations.

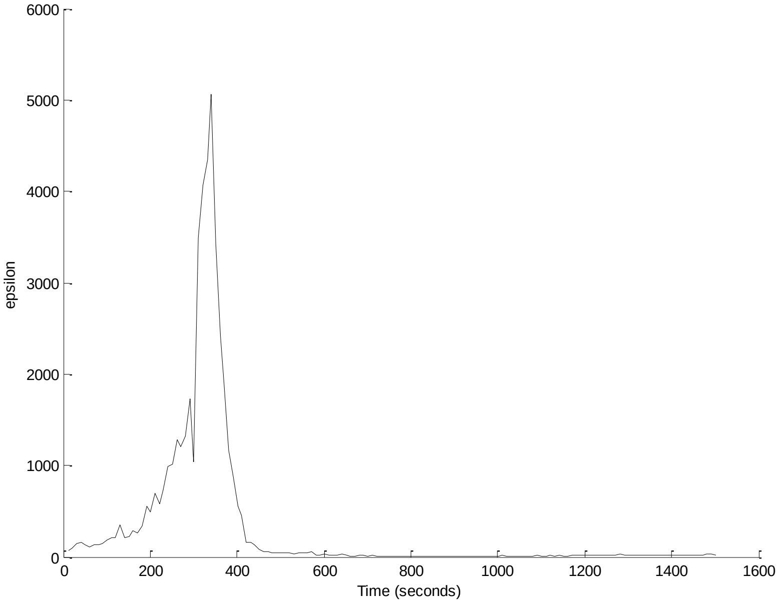


Figure 38. Normalized estimation error squared (ϵ) for LKF.

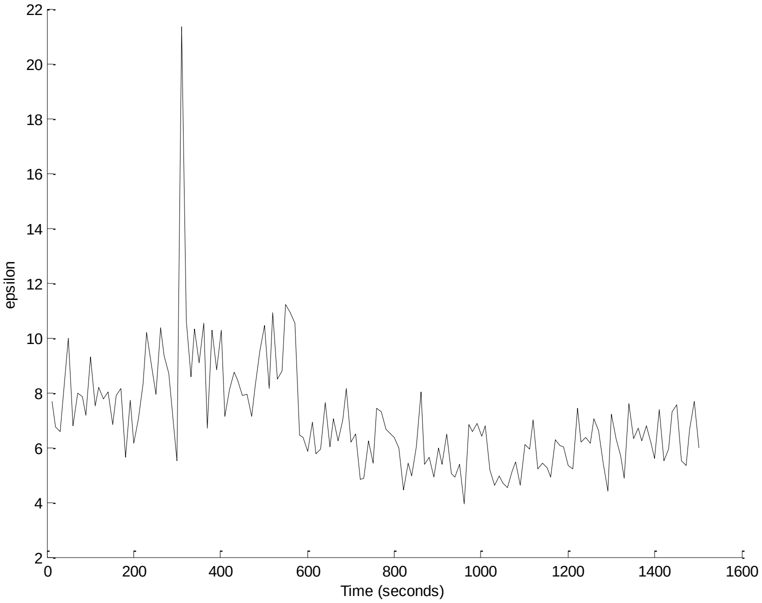


Figure 39. Normalized estimation error squared (ϵ) for EKF.

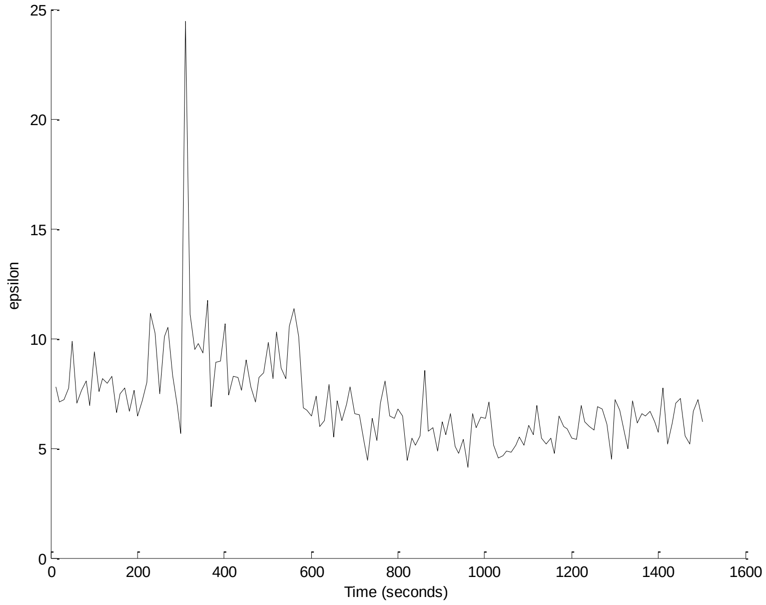


Figure 40. Normalized estimation error squared (ϵ) for UKF.

6.3. Test scenarios and validation process

Figure 37 provides a flow diagram of the MATLAB based program used to validate the bias correction and filtering algorithm. Selected MATLAB files used in the simulations are located in Appendix B. The algorithm was tested against eight scenarios involving three different target missile trajectories and a low earth orbit satellite trajectory. Each scenario involved two different sensor satellite trajectories. Both satellite trajectories were in a lead follower configuration at a 58 degree inclination at an altitude of 1300km and the lead satellite is 10 minutes ahead of the second. One trajectory was in an ascending orbit (away from the equator), the other was in a descending orbit (toward the equator). These different scenarios are described in Table 12. Graphical and numeric results of the individual scenarios are shown in Appendix A.

Each scenario test, with the exception of the Iridium satellite, included both bias cases as described in section 4.5 and used the bias EKF estimator. The Iridium scenario only tested the pure sinusoidal bias case. Table 13 lists the variables associated with all the test scenarios. For the EKF and UKF the individual sensor estimates were fused as described in Chapter 5.

Table 15. Target tracking scenarios

Scenario Name	Launch Point	Impact Point	Time of Flight
NEA_SF	North Korea	San Francisco	1800 sec
NEA_HI	North Korea	Hawaii	1500 sec
NEA_GU	North Korea	Guam	1000 sec
NEA_IR	LEO Iridium satellite		1800 sec

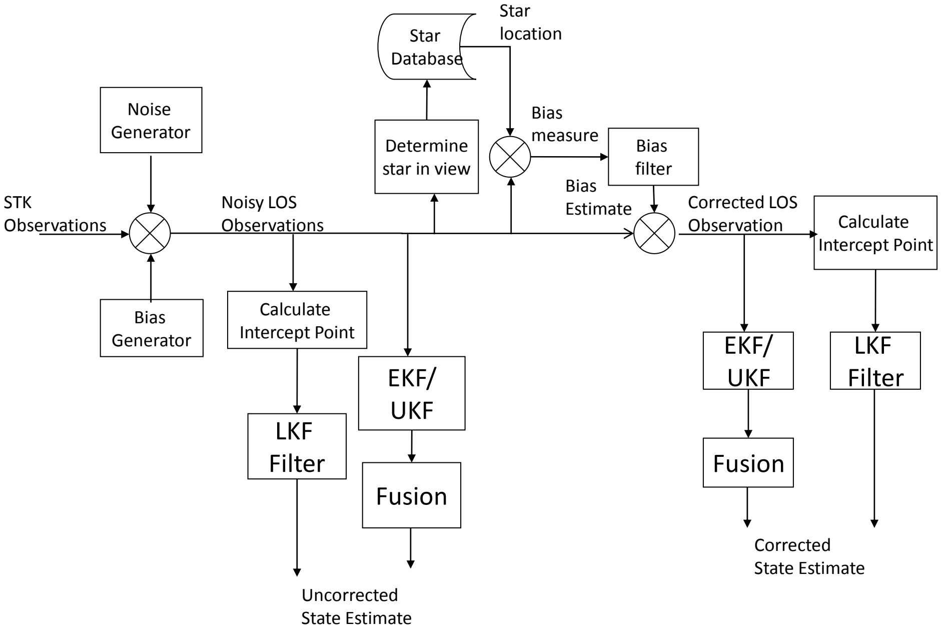


Figure 41. Bias correction test program flow chart

Table 16. Common test variables

Process noise (σ_p)	Observation error (σ_θ, σ_ϕ)	Bias noise error (σ_β)	Field of view	Time step (T_s)	Number of runs
10 m	10 μrad	2 μrad	0.8°	1 sec	25

In every scenario, the bias correction algorithm improved the tracking accuracy of the sensor system. This result was independent of target range, the time spent tracking the target, which varied from 1000 seconds to 1800 seconds, the portion of the celestial sphere traced out by the sensors, or the number of stars observed. In many cases the target flew under the sensors and the earth blocked any star observations. Since algorithm was designed to reset the nearest star table pointer after 60 seconds of track without a star detection, the system was able to re-acquire the appropriate star from which to obtain a bias error observation measurement. Furthermore, although we saw a difference in results between the two bias cases, this difference was minor since the bias estimator was able to closely estimate the bias characteristic even with gaps in star observations. The quality of the bias estimate was dependent on the number of star observations before a gap appeared.

To measure the overall performance of the three nonlinear filters with the bias correction algorithm we compared the aggregate percentage improvement of the target position error for all the scenarios and bias cases. As we can see in Table 14, there is only a slight difference between the filter types with the EKF being only slightly better. One can see that there is significant improvement in the target position estimation as a result of bias error correction and non-linear filtering for smaller values of bias error, but as bias error increases the effectiveness of the nonlinear filters is not as great.

Furthermore, the bias correction has a negative effect on error when there is no post correction filtering. This is likely due to the random sensor errors involved with both the star observation and the target observation combining to produce a larger error effect. In the case of larger bias errors, the small improvement between the different filters is most likely due to the fact that with the bias correction, the error is reduced significantly before filtering and the filter cannot provide much improvement.

Table 17. Average percent error improvement for all test scenarios

Bias (β_0)	No filter	LKF	EKF	UKF
None	-13.83	8.94	23.12	**24.08**
.02 *m*rad	31.90	45.09	**54.16**	53.20
.2 *m*rad	90.04	91.75	**92.67**	92.55
2 *m*rad	95.12	95.19	**95.25**	95.22

7. SUMMARY AND FUTURE RESEARCH DIRECTIONS

7.1. Dissertation Summary

In this study we set out to improve target tracking of a space-based Infrared tracking system through bias estimation from in-situ celestial observations and nonlinear estimation of the target state. The objective of the study was to focus the problem on solving bias issues in a space-based infrared tracking system based on the Missile Defense Agency's Space Tracking and Surveillance System.

We conducted the study in three phases. First we determined the magnitude of target position error for various amounts of bias in the system. Both spatial error in azimuth and elevation of the sensor, which contributed to line of sight errors, and in sampling time, where one sensor sampled the elevation and azimuth readings a short time after the first. It was discovered that the error in the observation line of sight contributed significantly to the location error of the target and that this error was proportional to the range from the target. The timing bias also contributed to target position error, but it was insignificant compared to the effect of the line of sight bias.

In the second phase of the study we developed a star detection and sorting algorithm that allowed the system to determine if a star would be seen in the sensor field of view. When a star was expected to appear, the algorithm used the difference of the measured star position and the expected star position to obtain a LOS bias measurement. After the sampling, we tested two different bias estimation techniques on two different bias waveforms to estimate bias values between star observations. This was necessary because there were periods of time when stars were not in the line of sight. The first used a simple sample and hold method. The second utilized an Extended Kalman Filter to estimate the sinusoidal bias vector. The EKF method provided better performance for the two types of bias waveforms tested. We also determined that the bias sampling and correction algorithm provided a substantial improvement in target location. Variable aperture sizes allowed us to see how the algorithm's performance depended upon the number of stars observed. We found that even with the smallest aperture size, the algorithm provided significant improvement in performance.

The aperture size of the sensor had a near linear relationship between the number of stars detected and the target mean error. Even with a sensor as small as .3 degrees radius there were ample stars detected and target error was within acceptable limits. This effect was confirmed in the multiple scenario testing where aperture size was set to .4 degrees radius and produced good mean error values for all the scenarios.

The third phase of the study compared the performance of three nonlinear estimation filters to provide a final state estimate. The first is a Linearized Kalman Filter that used the intercept point of the line of sight vectors to the target for the filter measurements. The second was a standard Extended Kalman Filter, and the third was an Unscented Kalman Filter. In the latter two filters a distributed fusion approach was used where a separate target estimate was calculated for each sensor. These estimates were then fused to provide a common target estimate. The performance of these nonlinear filters was compared to the Cramer-Rao lower bound and a measure of filter consistency, the normalized (state) estimation error squared. The filters tended to approach the PCLRB, but were generally a factor of 10-20 away from the bound. This is most likely due to the number of coordinate frame transformations adding additional error to the results. Additionally, it is known that the EKF will add bias to the state estimation, see [49]. In the test scenario there was little difference found in the performance of the three filters. However, the difference in consistency varied greatly from the LKF to the EKF and UKF. We found that the LKF was very inconsistent in the NEES measurement, yet the EKF and UKF were very close to the expected value.

After the algorithm development phases, in the final phase we ran tracking simulations against a number of different scenarios involving different ballistic missile and space target trajectories using different sensor satellite orbits. In all cases, the bias correction algorithm showed significant improvement in target tracking with as much as 95 percent improvement in the worst scenarios. Of the three nonlinear filters, the EKF slightly out-performed the UKF and LKF. However, in many scenarios, when there was no bias input into the line of sight measurements, the bias correction algorithm performed worse than if there was no bias correction. This was more consistently true when using the LKF estimator.

7.2. Implications of the results

The results of this study demonstrate the effectiveness of utilizing stellar observations for bias correction when tracking ballistic targets with space-based IR sensors. We showed that it did not matter where the bias occurs - in the sensor pointing, spacecraft attitude, or the measurement timing – that without bias correction, position error overwhelmed the ability of a nonlinear filter to make an adequate estimate of the target state. Furthermore, the greater the bias, the more significant the impact on the sensor's ability to discern the target position from the noisy observations.

The ability to correct target observations with in-situ observations while tracking increases the effectiveness of pre-tracking calibration by accounting for slowly varying bias as a result of spacecraft thermal distortion. From the findings of this study, we conclude that in-situ celestial observations made during the tracking of a target will provide a superior method of bias correction and improve tracking accuracy. We also found that there was little difference between the EKF and UKF in improving the target state estimate. The estimation filters used in the study all approached the PCRLB and proved adequate in providing an estimate of the target state

Although the accuracy of the target estimates depended upon the number of stars detected, we found that even with a small sensor aperture size there were sufficient measurements to provide an estimate of the bias using both the sample and hold method or the more sophisticated, yet very accurate, EKF method. The accuracy of the bias estimate using the EKF led to better results in both cases of the bias models used in the study.

We also adapted a unique star search algorithm to rapidly determine if a star was within the field of view. This allowed for measuring bias in each time step if there were a star available. The algorithm also enabled us to locate stars quickly when the sensor was shielded by the earth or encountered sunlight and moonlight.

7.3. Contributions

The results of this study are consistent with the previous work done in this area. In this study we extended currently developed techniques of state estimation to a new problem of space-based IR tracking of a ballistic missile. We showed how background stars can provide a means for measuring and correcting bias in a space-based IR tracking system. Most importantly we showed how target location significantly improved from the uncorrected state. Even when bias is relatively small, the improvement in target tracking accuracy is on the order of tens of percent. Earlier versions of error correction only allowed for calibration prior to the start of target tracking, or during the tracking phase when the sensor had to break track and to calibrate. This old method fails to account for any bias fluctuations that occur following the sensor calibration. The new method shown here enables real-time bias correction while the target is in track, correcting for any errors that result from thermal or dynamic effects on the system.

A second contribution from this work is the formulation of a separate bias filter to estimate the bias error from the non-synchronous measurements. Although other researched showed how one can use a separate bias filter to estimate the bias, those methods were restricted to linear processes, and did not use a model of the bias to accurately estimate the bias in between observations. Our method more accurately models the bias and uses a nonlinear EKF to estimate the bias waveform accurately. This technique enables better bias estimates during periods when no stars are observable.

Another contribution of this study was the comparison of the nonlinear filters to the target estimation problem. Here we used three different methods of target location. The first was using the intercept point of two vectors coupled with a LKF. Two new techniques were developed for using the LKF for target state estimation. The first was in calculating an estimated intercept point from the two target line of sight vectors. The second was estimating the range variance from the intercept geometry associated with the target line of sight and their measurement variances. We were unable to find any use of this methodology in the literature. Although it did not significantly improve the state estimate over the EKF or UKF methodologies, it did provide a simpler method. The study also showed that the other two methods, the EKF and UKF were not appreciably different in their effectiveness.

A final contribution of our work was the development of a fast search algorithm based on the use of the nearest neighbor table to this problem of sensor calibration. Previous work had applied this technique for rapid identification of stars for spacecraft attitude control [43]. This study extended the methodology to allow rapid determination if a star was within a field of view.

7.4. Study limitations

The obvious limitations associated with this work, was the simulative nature of using a MATLAB program with data provided from the STK simulation. Several assumptions and simplifications were required to replicate the actual conditions expected. For instance, the bias model, although representative of real world conditions, may not exactly match the conditions found in the spacecraft. Similarly, the absence of actual measurements provides a much “cleaner” picture than real data would allow. This is especially true with the assumption that there was only one target object and that track association issues were not considered in the study. Finally, we must mention that some simplifications and approximations were made to avoid security classification issues associated with the actual system hardware characteristics that could also affect the fidelity of the results.

7.5. Recommendations for Future Research

Although the derivations, analysis, and results of this study captured issues associated with improving the target tracking capability of a space based IR sensor using background stars and advanced estimation algorithms, a number of areas remain for future research. The analyses here established the theoretical understanding of the algorithm's performance and related issues but were subject to the limitations discussed above. To fully understand the capability offered here requires extending these findings into more detailed and sophisticated real life environments.

A first area to explore would be to extend the results to include multiple targets and false alarms and to increase the noise and uncertainty of target detection. False alarms would make identification of a star more complicated. To help resolve this problem, the algorithm may be adjusted to measure a star's intensity and compare it with the intensity found in the star catalog to aid in matching. Additionally, multiple targets would require the algorithm to perform data association hypothesis testing such as the nearest neighbor or Multiple Hypothesis Tests. This area of research would be of great benefit for Space Situational Awareness where many closely-spaced objects require identification and tracking. Bias removal could be critical to the success of the process. Bias removal in the track-to-track association problem is explored by Ferry in [50]. The algorithm could also be tested on maneuvering targets and targets that are lost and require reacquisition. The models tested here assumed continuous tracking of a target only under the influence of gravity. A maneuvering warhead or satellite may make the algorithm less effective. Also, widely spaced multiple targets in a mass missile attack or in space tracking would require the sensor to break track on one object and look at another.

Another area that may require further development is to include multiple bias models in the bias estimate. In this study we assumed only one model present at a time. A more inclusive set of models would use at least the two models explored here plus possibly a steady state bias where the sinusoidal component is zero. A direction that this research might take is to input the bias measurements into a separate Multiple Mixture Model estimator containing the different bias models [51] and [52]. Appling this technique would account for different effects of the space environment on the spacecraft and sensor.

A significant extension of this work would include adding the attitude control system into the model and perform simultaneous nonlinear estimation of sensor and attitude control bias using celestial observations. This would provide a fully integrated system along the lines presented by Wu in [39] but with simultaneous tracking. The correction of errors in the gyro bias from the observations would assist in ensuring accurate attitude references for coordinate frame transformation and spacecraft navigation.

Most desirable would be the use of data from the actual STSS system to validate the true nature of the bias error and to test the performance of the algorithm. The STSS system is now in orbit undergoing post-launch testing. It will acquire its first missile tracking data from a test scheduled in 2010. Actual data will allow the algorithm to be tested with false alarms and ghost tracks, multiple targets, and detected stars that are not in the Aura catalog. Initial testing could be done with this algorithm, but would probably need to be simplified and optimized for real time operation. This optimization would be especially necessary when trying to start multiple targets that might be generated from a target missile deploying decoys or rocket associated debris such as payload fairings and tanks.

Lastly, although not directly related to bias correction, since the STSS is only one of the systems in the missile defense and space tracking architecture, fusing the space based sensor tracking results with other sensor measurements is an area of interest worth pursuing. This could take the form of other sensors providing observations to help refine the space sensor track estimates, especially if there was missing observations, or vice versa. With only a few sensors in orbit, the STSS may not be present during the entire flight of a threat ballistic missile. Being able to receive a cue from other sensors is an important capability. Bias estimates from other systems also justify a bias removal methodology in either a centralized or distributed manner. This concept of estimation of bias errors of active and passive sensors used in connection with multisensory multitarget tracking is discussed by Sviestins in [53].

APPENDIX A: SIMULATION RESULTS

Scenario: Northeast Asia to San Francisco development scenario

This scenario utilizes two sensor satellites in a 600 nm altitude 60 degree inclination ascending node orbit. The target missile is an ICBM class missile launched from Northeast Asia targeting San Francisco. The missile flight time is 30 minutes. Figure A-1 is a diagram of the sensor satellites and target trajectories. Table A-1 provides numerical results for the scenario with sensor bias σ_S = 10 μrad. Table A-2 provides numerical results for the scenario with sensor bias σ_S = 50 μrad and σ_S = 100 μrad. Table A-3 provides results for the scenario with time step T_s = 5 sec and 10 sec.

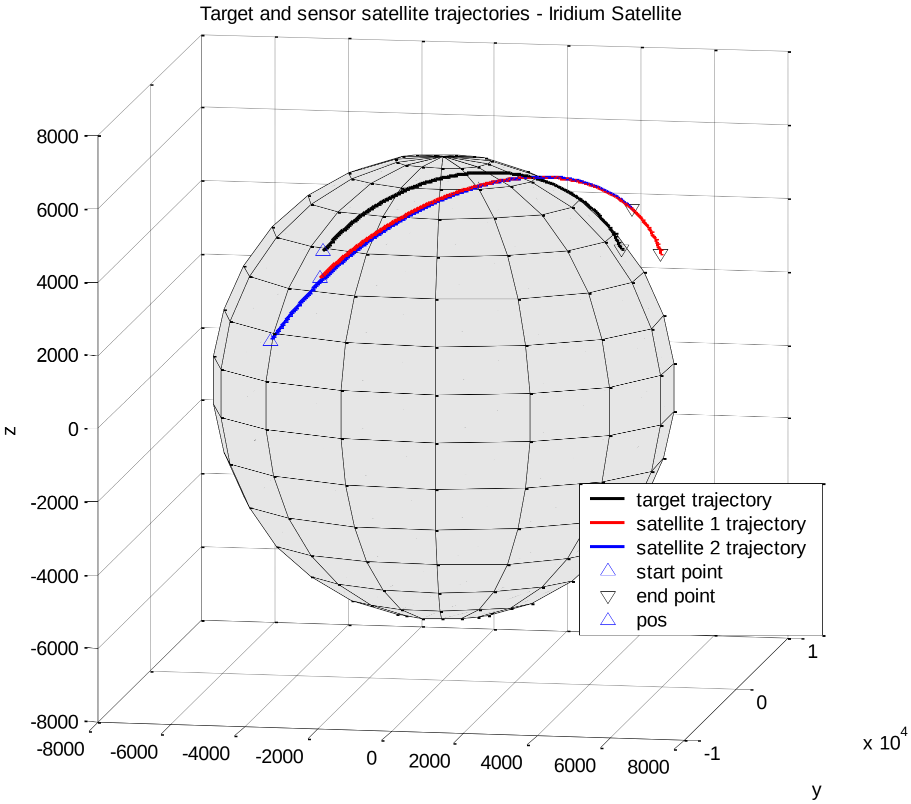


Figure A-1 Satellite and target trajectories for SEA-SF-development scenario.

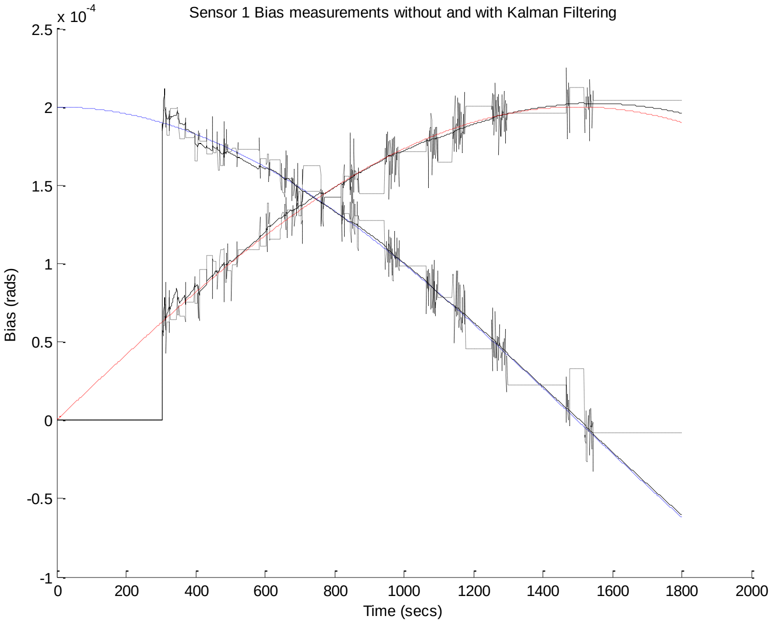


Figure A-2. Case 1: Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias

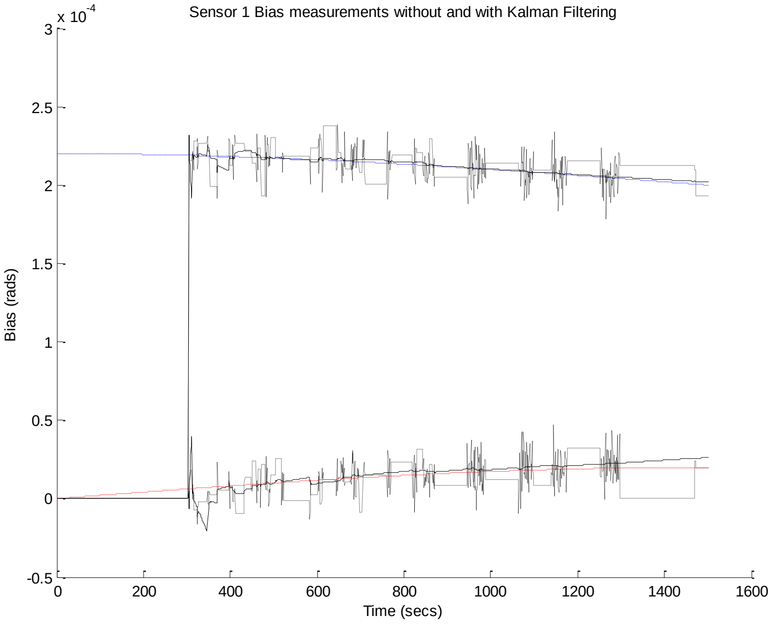


Figure A-3. Case 2: Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias

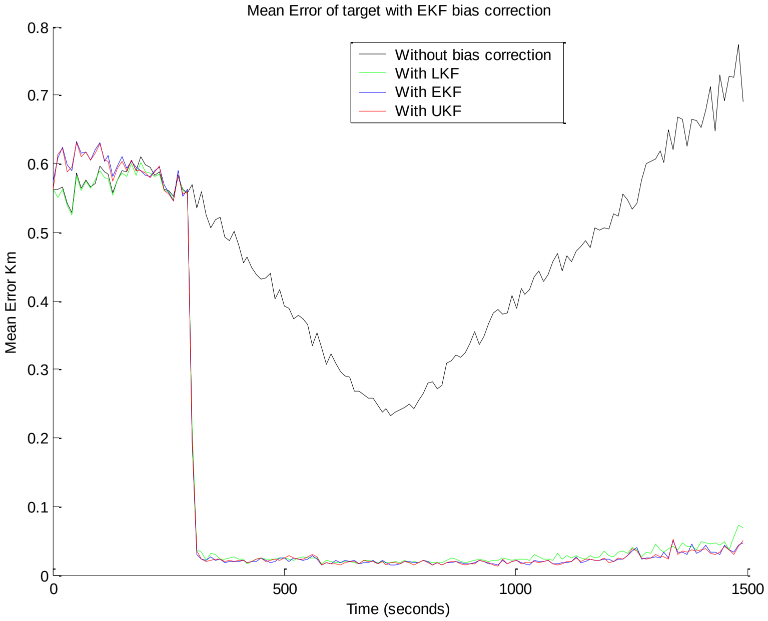


Figure A-4. Case 1: Mean error of target position for sinusoidal bias (β_0 = .2 *m*rads)

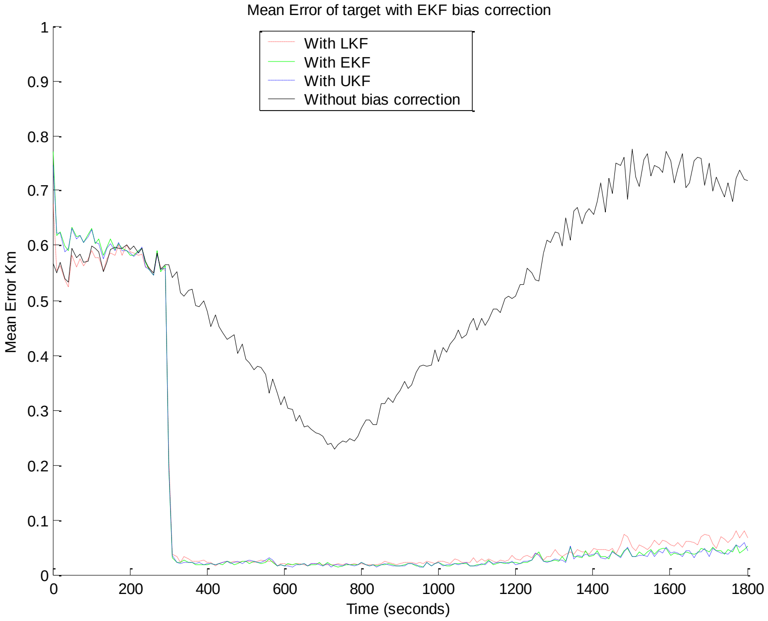


Figure A-5. Case 1: Mean error of target position for sinusoidal bias (β_0 = .2 *m*rads)

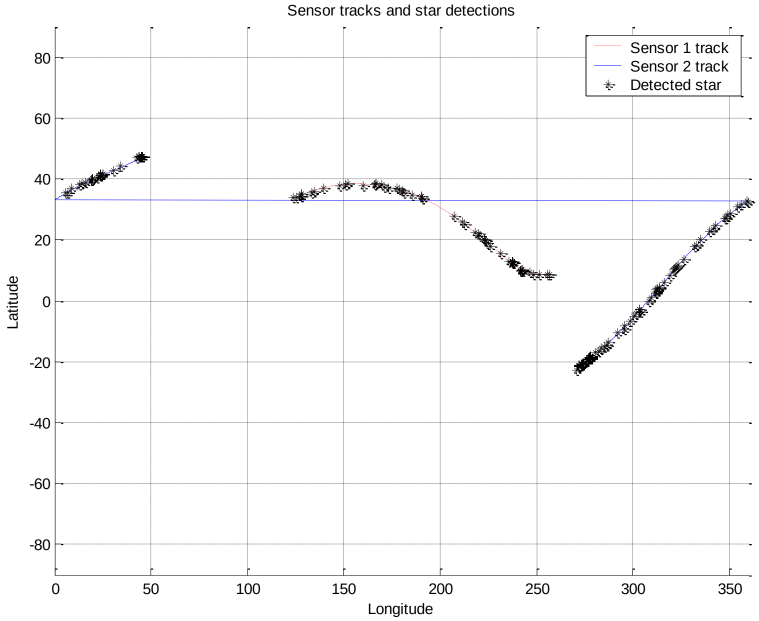


Figure A-6. Sensor trajectories with detected stars for NEA-SF-development scenario.

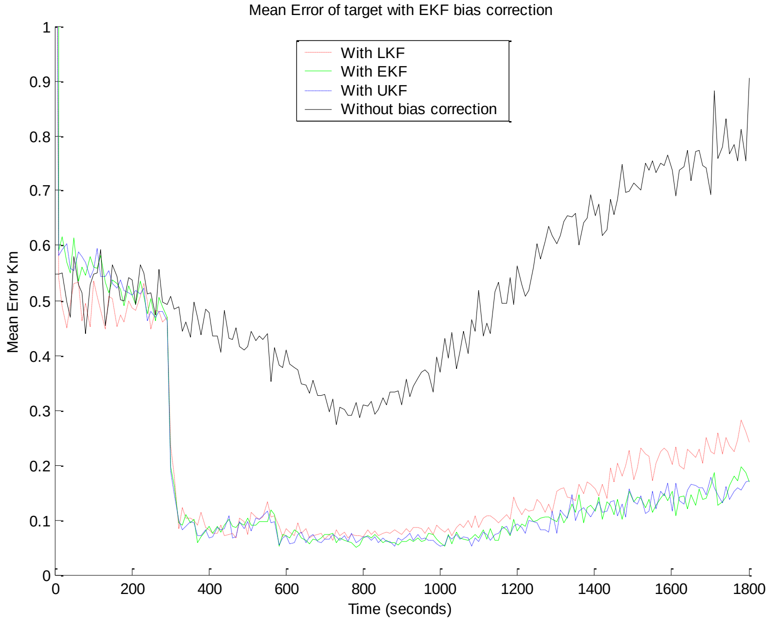


Figure A-7. Mean error of target position for sensor error (σ_s) = 50 μrad with sinusoidal bias (β_0 = .2 *m*rads)\

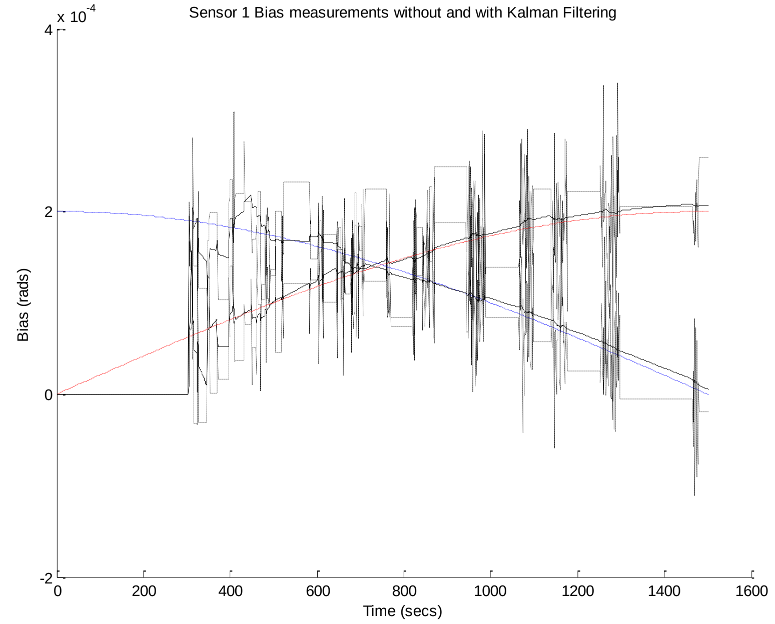


Figure A-8. Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias and sensor error ($\sigma_{\theta,\phi}$) = 50 μrad

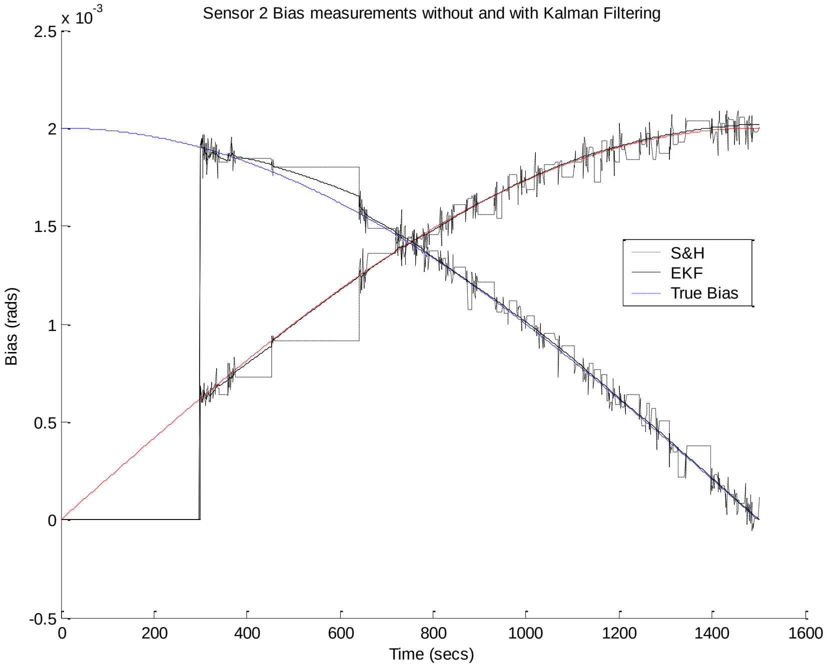


Figure A-9. Sensor 2 bias measurements and estimates with and without bias EKF for pure sinusoidal bias and sensor error ($\sigma_{\theta,\phi}$) = 50 μrad

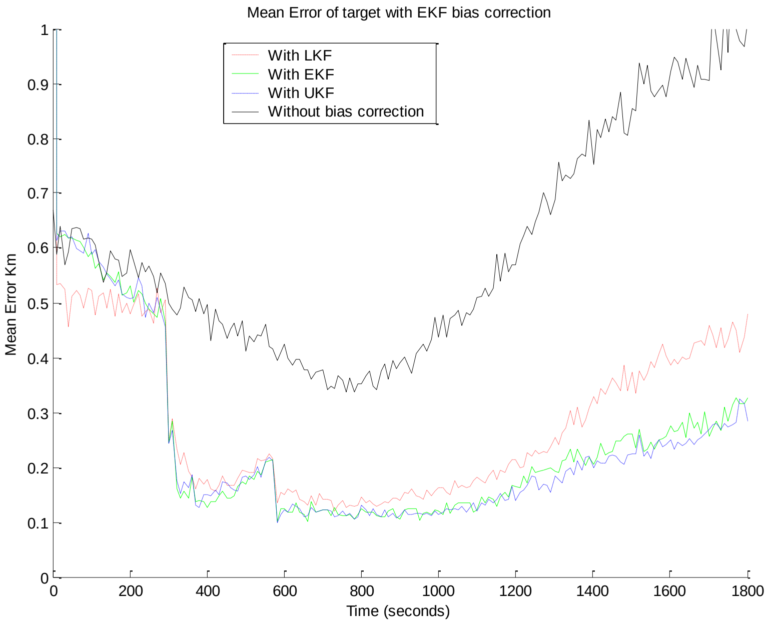


Figure A-10. Mean error of target position for sensor error (σ_s) = 100 μrad with sinusoidal bias (β_0 = .2 $m$rads)

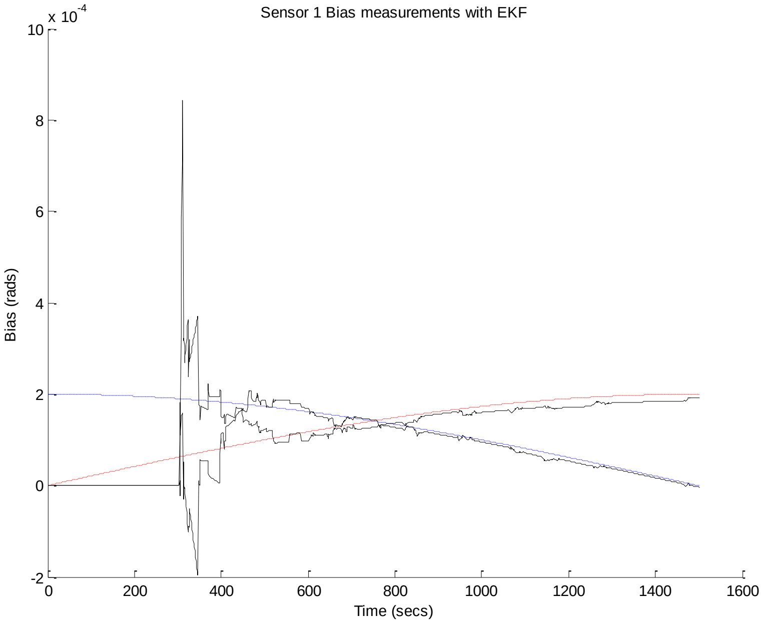


Figure A-11. Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias and sensor error ($\sigma_{\theta,\phi}$) = 100 μrad

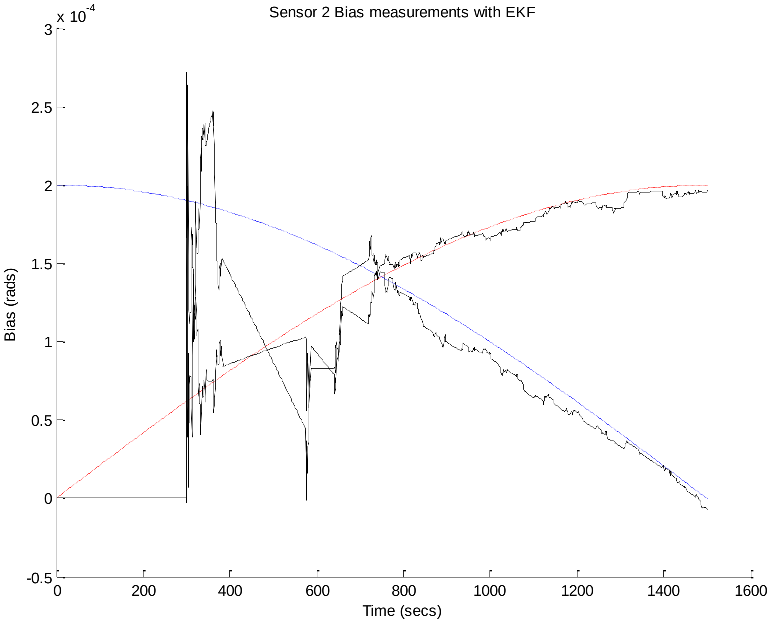


Figure A-12. Sensor 2 bias measurements and estimates with and without bias EKF for pure sinusoidal bias and sensor error ($\sigma_{\theta,\phi}$) = 100 μrad

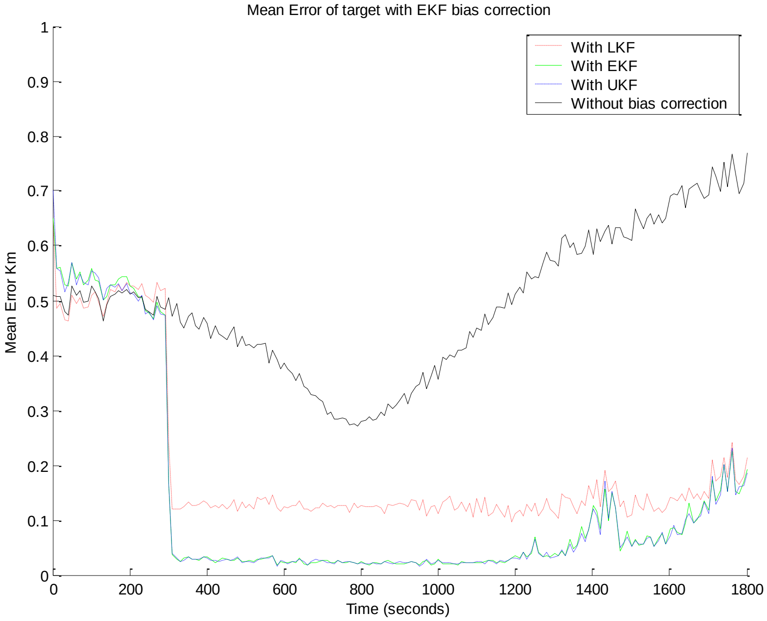


Figure A-13. Mean error of target position for time step (T_s) = 5 seconds with sinusoidal bias (β_0 = .2 *m*rads)\

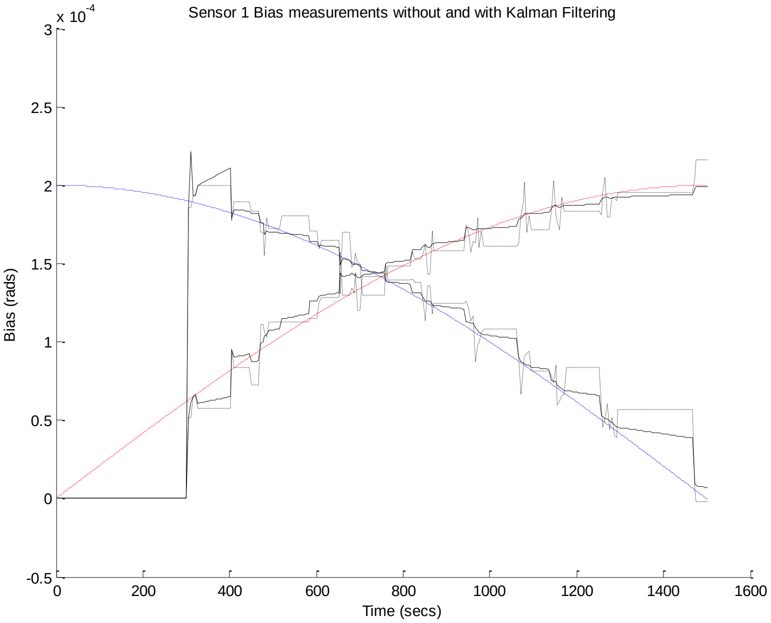


Figure A-14. Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias and time step (T_s) = 5 seconds.

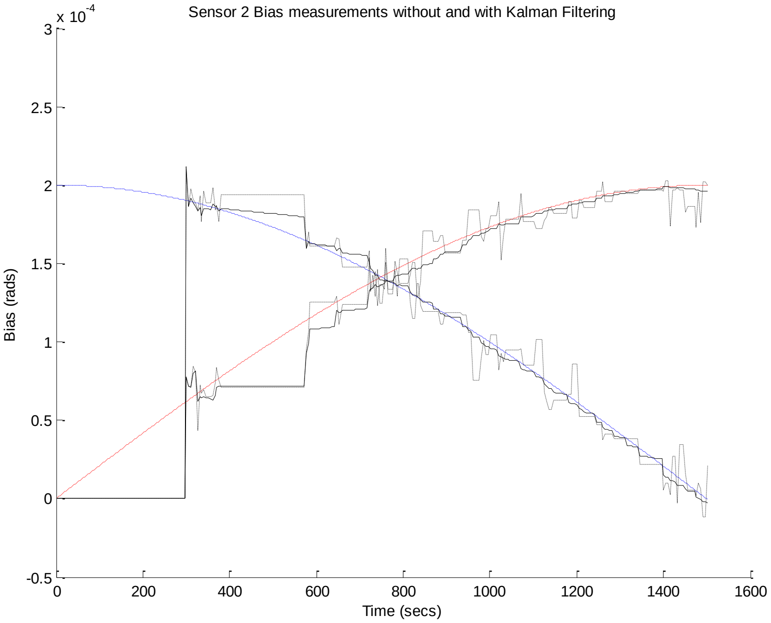


Figure A-15. Sensor 2 bias measurements and estimates with and without bias EKF for pure sinusoidal bias and time step (T_s) = 5 seconds.

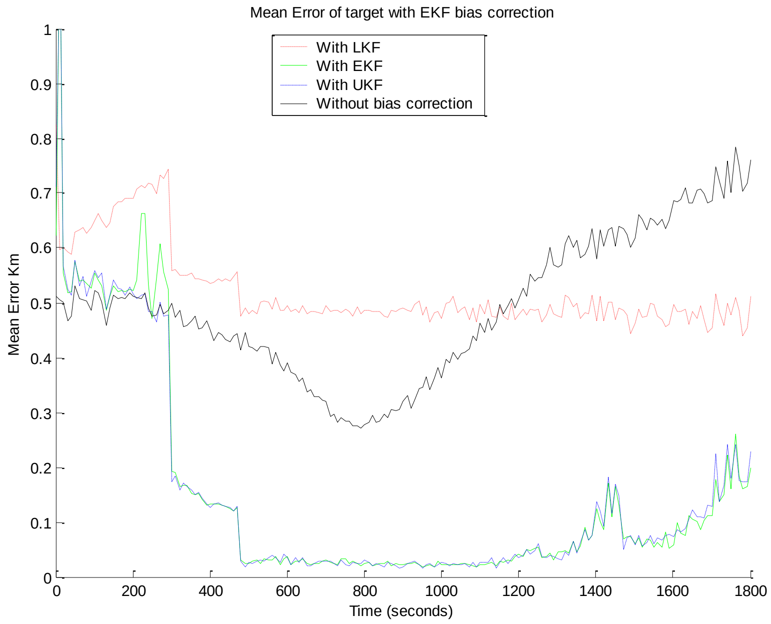


Figure A-16. Mean error of target position for time step (T_s) = 10 seconds with sinusoidal bias (β_0 = .2 *m*rads).

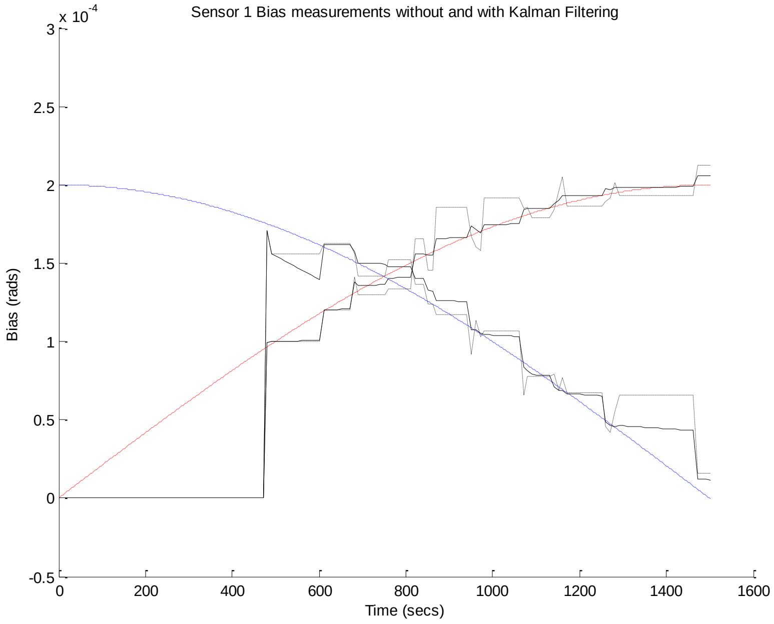


Figure A-17. Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias and time step (T_s) = 10 seconds.

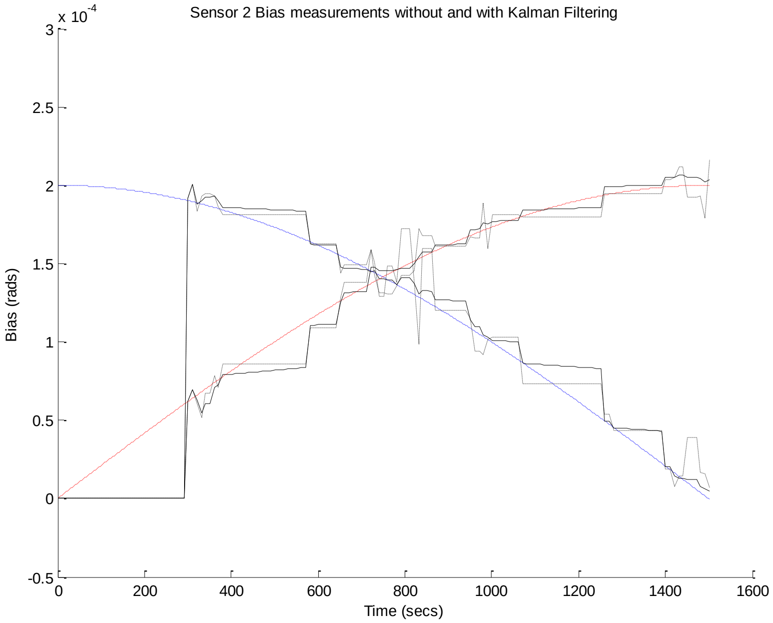


Figure A-18. Sensor 2 bias measurements and estimates with and without bias EKF for pure sinusoidal bias and time step (T_s) = 10 seconds.

NEA-SF-DEVELOP Case 1														
Filter type		**without bias correction**		**with EKF bias correction**			**with Sample and Hold correction**			**With EKF bias correction and post filtering**			**Numbe of stars**	
	bias	**Mean Err**	**STD**	**Mean Err**	**STD**	**Pct Impr**	**Mean Err**	**STD**	**Pct Impr**	**Mean Err**	**STD**	**Pct Impr**	**Sensor 1**	**Sensor 2**
LKF	0	4.06E-02	5.67E-04	4.10E-02	7.97E-04	-1.17	4.77E-02	4.14E-03	-17.74	3.20E-02	9.94E-04	21.18	408	800
	2.00E-05	6.21E-02	6.69E-04	4.11E-02	7.81E-04	33.90	4.62E-02	3.51E-03	25.68	3.20E-02	7.06E-04	48.42	406	801
	2.00E-04	4.89E-01	5.62E-04	4.18E-02	8.20E-04	91.44	6.53E-02	8.61E-03	86.63	3.35E-02	8.25E-04	93.14	412	799
	2.00E-03	4.88E+00	7.95E-04	5.53E-02	9.32E-04	98.87	3.74E-01	1.27E-02	92.33	5.06E-02	1.04E-03	98.96	466	795
EKF	0.00E+00	4.06E-02	5.98E-04	4.11E-02	7.04E-04	-1.11	4.69E-02	3.68E-03	-15.64	2.56E-02	8.22E-04	36.87	406	800
	2.00E-05	6.21E-02	7.04E-04	4.12E-02	7.38E-04	33.59	4.72E-02	3.05E-03	23.96	2.58E-02	8.45E-04	58.45	407	801
	2.00E-04	4.89E-01	6.24E-04	4.18E-02	8.12E-04	91.44	6.42E-02	6.85E-03	86.87	2.71E-02	6.19E-04	94.46	413	800
	2.00E-03	4.88E+00	6.11E-04	5.53E-02	9.29E-04	98.87	3.72E-01	1.06E-02	92.37	4.41E-02	1.22E-03	99.10	468	797
UKF	0.00E+00	4.07E-02	5.33E-04	4.11E-02	6.14E-04	-0.86	4.68E-02	3.47E-03	-14.94	2.56E-02	7.40E-04	37.01	406	801
	2.00E-05	6.20E-02	6.38E-04	4.08E-02	6.80E-04	34.14	4.58E-02	3.50E-03	26.16	2.54E-02	6.99E-04	58.95	407	799
	2.00E-04	4.89E-01	6.71E-04	4.19E-02	8.25E-04	91.42	6.27E-02	7.25E-03	87.17	2.69E-02	7.45E-04	94.49	412	799
	2.00E-03	4.88E+00	7.64E-04	5.51E-02	8.82E-04	98.87	3.73E-01	1.07E-02	92.35	4.39E-02	1.06E-03	99.10	467	794
NEA-SF-DEVELOP Case 2														
LKF	0.00E+00	3.25E-02	4.46E-04	3.29E-02	6.32E-04	-1.44	3.58E-02	2.45E-03	-10.25	2.65E-02	7.21E-04	18.43	406	801
	2.00E-05	5.34E-02	6.42E-04	3.31E-02	8.18E-04	38.10	3.63E-02	2.66E-03	32.14	2.66E-02	8.50E-04	50.22	406	801
	2.00E-04	4.45E-01	7.22E-04	3.40E-02	9.16E-04	92.35	3.70E-02	2.46E-03	91.67	2.78E-02	1.04E-03	93.76	397	790
	2.00E-03	4.44E+00	7.70E-04	4.94E-02	1.46E-03	98.89	4.67E-02	3.38E-03	98.95	4.61E-02	1.51E-03	98.96	383	768
EKF	0.00E+00	3.25E-02	5.62E-04	3.30E-02	7.23E-04	-1.33	3.63E-02	3.21E-03	-11.60	2.18E-02	9.19E-04	32.88	409	800
	2.00E-05	5.35E-02	6.55E-04	3.31E-02	7.53E-04	38.16	3.60E-02	2.60E-03	32.62	2.20E-02	1.04E-03	58.90	405	801
	2.00E-04	4.44E-01	7.10E-04	3.37E-02	7.84E-04	92.41	3.75E-02	3.13E-03	91.56	2.33E-02	1.08E-03	94.76	397	792
	2.00E-03	4.44E+00	8.50E-04	4.93E-02	1.35E-03	98.89	4.70E-02	3.48E-03	98.94	4.43E-02	1.69E-03	99.00	383	770
UKF	0.00E+00	3.25E-02	4.55E-04	3.30E-02	6.48E-04	-1.26	3.59E-02	2.59E-03	-10.35	2.18E-02	8.32E-04	33.14	406	801
	2.00E-05	5.34E-02	7.04E-04	3.30E-02	8.93E-04	38.20	3.62E-02	2.83E-03	32.15	2.18E-02	9.24E-04	59.20	405	800
	2.00E-04	4.45E-01	5.73E-04	3.38E-02	8.37E-04	92.40	3.68E-02	2.35E-03	91.71	2.32E-02	9.80E-04	94.77	397	791
	2.00E-03	4.44E+00	6.22E-04	4.93E-02	1.22E-03	98.89	4.68E-02	3.15E-03	98.95	4.41E-02	1.48E-03	99.01	383	767

Table A-1. Performance results from NEA-SF-development. Number of runs = 50, Field of view = 1.0^{o}, Start time 0 s, End time 1800 s

NEA-SF-DEVELOP sensor bias = 50 urad														
Filter type		**without bias correction**		**with EKF bias correction**			**with Sample and Hold correction**			**With EKF bias correction and post filtering**			**Numbe of stars**	
	bias	**Mean Err**	**RMS Err**	**Mean Err**	**RMS Err**	**Pct Impr**	**Mean Err**	**RMS Err**	**Pct Impr**	**Mean Err**	**RMS Err**	**Pct Impr**	**Sensor 1**	**Sensor 2**
LKF	0	1.45E-01	1.62E-01	1.48E-01	1.66E-01	-2.18	1.60E-01	1.81E-01	-10.41	1.03E-01	1.14E-01	28.98	410	798
	2.00E-05	1.50E-01	1.67E-01	1.48E-01	1.66E-01	0.94	1.57E-01	1.77E-01	-4.75	1.04E-01	1.15E-01	30.55	412	798
	2.00E-04	4.55E-01	4.67E-01	1.49E-01	1.67E-01	67.20	1.54E-01	1.74E-01	66.09	1.04E-01	1.14E-01	77.23	412	802
	2.00E-03	4.37E+00	4.37E+00	1.56E-01	1.74E-01	96.43	2.57E-01	2.79E-01	94.12	1.17E-01	1.28E-01	97.32	470	793
EKF	0.00E+00	1.45E-01	1.62E-01	1.49E-01	1.66E-01	-2.44	1.60E-01	1.81E-01	-10.27	8.37E-02	9.41E-02	42.39	403	802
	2.00E-05	1.52E-01	1.69E-01	1.48E-01	1.66E-01	2.29	1.61E-01	1.81E-01	-5.90	8.29E-02	9.34E-02	45.29	405	802
	2.00E-04	4.56E-01	4.67E-01	1.49E-01	1.67E-01	67.22	1.64E-01	1.86E-01	63.94	8.50E-02	9.57E-02	81.35	408	797
	2.00E-03	4.37E+00	4.37E+00	1.54E-01	1.72E-01	96.47	2.55E-01	2.76E-01	94.16	1.01E-01	1.12E-01	97.69	469	794
UKF	0.00E+00	1.46E-01	1.63E-01	1.48E-01	1.66E-01	-1.70	1.58E-01	1.79E-01	-8.80	8.13E-02	9.16E-02	44.14	408	803
	2.00E-05	1.52E-01	1.70E-01	1.48E-01	1.65E-01	2.66	1.57E-01	1.77E-01	-2.97	8.11E-02	9.13E-02	46.68	409	800
	2.00E-04	4.55E-01	4.67E-01	1.48E-01	1.65E-01	67.54	1.60E-01	1.82E-01	64.96	8.33E-02	9.35E-02	81.70	415	799
	2.00E-03	4.37E+00	4.37E+00	1.55E-01	1.73E-01	96.45	2.50E-01	2.70E-01	94.28	1.00E-01	1.11E-01	97.71	465	794
NEA-SF-DEVELOP sensor bias = 100 urad														
LKF	0.00E+00	3.62E-01	4.07E-01	3.68E-01	4.14E-01	-1.43	4.25E-01	4.84E-01	-17.16	2.37E-01	2.60E-01	34.69	410	804
	2.00E-05	3.65E-01	4.10E-01	3.67E-01	4.12E-01	-0.57	4.13E-01	4.71E-01	-13.40	2.35E-01	2.58E-01	35.42	407	800
	2.00E-04	5.94E-01	6.37E-01	3.69E-01	4.15E-01	37.93	4.23E-01	4.83E-01	28.83	2.41E-01	2.64E-01	59.39	409	800
	2.00E-03	4.89E+00	4.89E+00	3.77E-01	4.24E-01	92.28	5.69E-01	6.32E-01	88.36	2.64E-01	2.87E-01	94.59	467	799
EKF	0.00E+00	3.61E-01	4.06E-01	3.67E-01	4.12E-01	-1.45	4.25E-01	4.84E-01	-17.57	1.79E-01	2.01E-01	50.45	405	801
	2.00E-05	3.66E-01	4.11E-01	3.69E-01	4.14E-01	-0.74	4.26E-01	4.85E-01	-16.44	1.80E-01	2.02E-01	50.73	413	797
	2.00E-04	5.97E-01	6.41E-01	3.67E-01	4.13E-01	38.51	4.27E-01	4.89E-01	28.52	1.78E-01	2.00E-01	70.13	407	806
	2.00E-03	4.89E+00	4.89E+00	3.77E-01	4.24E-01	92.29	5.72E-01	6.34E-01	88.29	2.06E-01	2.29E-01	95.79	471	795
UKF	0.00E+00	3.64E-01	4.08E-01	3.68E-01	4.13E-01	-1.08	4.21E-01	4.78E-01	-15.89	1.71E-01	1.92E-01	52.95	408	798
	2.00E-05	3.64E-01	4.10E-01	3.65E-01	4.11E-01	-0.23	4.21E-01	4.81E-01	-15.72	1.69E-01	1.89E-01	53.69	408	791
	2.00E-04	5.98E-01	6.41E-01	3.69E-01	4.15E-01	38.23	4.18E-01	4.77E-01	29.99	1.71E-01	1.91E-01	71.41	411	799
	2.00E-03	4.89E+00	4.89E+00	3.74E-01	4.20E-01	92.35	5.88E-01	6.49E-01	87.97	1.97E-01	2.18E-01	95.96	469	790

Table A-2. Performance results from NEA-SF-development with increased sensor bias. Number of runs = 50, Field of view = 1.0°.

NEA-SF-DEVELOP Time step = 5 sec														
Filter type		without bias correction		with EKF bias correction			with Sample and Hold correction			With EKF bias correction and post filtering			Numbe of stars	
	bias	Mean Err	RMS Err	Mean Err	RMS Err	Pct Impr	Mean Err	RMS Err	Pct Impr	Mean Err	RMS Err	Pct Impr	Sensor 1	Sensor 2
LKF	0	4.02E-02	4.51E-02	4.10E-02	4.59E-02	-1.97	4.51E-02	5.08E-02	-12.32	1.27E-01	1.29E-01	-215.35	81	157
	2.00E-05	6.18E-02	6.61E-02	4.35E-02	4.82E-02	29.71	4.88E-02	5.43E-02	21.16	1.27E-01	1.29E-01	-105.79	80	157
	2.00E-04	4.90E-01	4.91E-01	1.17E-01	1.22E-01	76.08	1.24E-01	1.29E-01	74.69	1.85E-01	1.88E-01	62.15	81	159
	2.00E-03	4.89E+00	4.89E+00	9.58E-01	9.60E-01	80.40	1.01E+00	1.01E+00	79.32	1.00E+00	1.01E+00	79.46	96	152
EKF	0.00E+00	4.06E-02	4.53E-02	4.14E-02	4.62E-02	-1.88	4.47E-02	5.02E-02	-10.12	3.69E-02	4.22E-02	9.09	80	157
	2.00E-05	6.20E-02	6.63E-02	4.35E-02	4.85E-02	29.86	4.72E-02	5.26E-02	23.81	3.90E-02	4.43E-02	37.05	80	157
	2.00E-04	4.90E-01	4.90E-01	1.17E-01	1.21E-01	76.21	1.22E-01	1.27E-01	75.05	1.13E-01	1.18E-01	76.91	81	157
	2.00E-03	4.89E+00	4.89E+00	9.58E-01	9.60E-01	80.41	1.01E+00	1.01E+00	79.34	9.57E-01	9.60E-01	80.43	96	151
UKF	0.00E+00	4.02E-02	4.51E-02	4.10E-02	4.60E-02	-2.05	4.64E-02	5.24E-02	-15.58	3.52E-02	4.05E-02	12.27	80	157
	2.00E-05	6.18E-02	6.61E-02	4.36E-02	4.84E-02	29.43	4.85E-02	5.37E-02	21.51	3.81E-02	4.32E-02	38.25	80	158
	2.00E-04	4.90E-01	4.91E-01	1.18E-01	1.22E-01	76.03	1.25E-01	1.31E-01	74.49	1.14E-01	1.18E-01	76.81	81	158
	2.00E-03	4.89E+00	4.89E+00	9.59E-01	9.61E-01	80.39	1.01E+00	1.01E+00	79.38	9.58E-01	9.60E-01	80.41	96	151
NEA-SF-DEVELOP Time step = 10 sec														
LKF	0.00E+00	4.07E-02	4.57E-02	4.23E-02	4.73E-02	-3.83	4.51E-02	5.07E-02	-10.77	4.81E-01	4.81E-01	-1080.94	34	75
	2.00E-05	6.13E-02	6.56E-02	4.62E-02	5.09E-02	24.63	4.96E-02	5.49E-02	19.08	4.81E-01	4.81E-01	-685.00	32	75
	2.00E-04	4.90E-01	4.91E-01	1.41E-01	1.45E-01	71.27	1.46E-01	1.50E-01	70.31	5.19E-01	5.20E-01	-5.84	32	75
	2.00E-03	4.89E+00	4.89E+00	1.22E+00	1.22E+00	75.10	1.25E+00	1.25E+00	74.41	1.47E+00	1.47E+00	69.93	43	70
EKF	0.00E+00	4.11E-02	4.58E-02	4.24E-02	4.73E-02	-3.39	4.62E-02	5.21E-02	-12.63	4.77E-02	5.81E-02	-16.31	32	75
	2.00E-05	6.23E-02	6.66E-02	4.58E-02	5.06E-02	26.43	4.98E-02	5.52E-02	19.95	5.05E-02	6.08E-02	18.87	32	75
	2.00E-04	4.90E-01	4.91E-01	1.38E-01	1.42E-01	71.76	1.42E-01	1.46E-01	71.11	1.39E-01	1.46E-01	71.65	32	74
	2.00E-03	4.90E+00	4.90E+00	1.22E+00	1.22E+00	75.12	1.25E+00	1.25E+00	74.48	1.23E+00	1.23E+00	74.97	43	70
UKF	0.00E+00	4.07E-02	4.58E-02	4.21E-02	4.73E-02	-3.38	4.60E-02	5.19E-02	-13.02	3.71E-02	4.27E-02	8.93	32	75
	2.00E-05	6.16E-02	6.59E-02	4.61E-02	5.10E-02	25.14	5.01E-02	5.55E-02	18.63	4.12E-02	4.65E-02	33.02	32	76
	2.00E-04	4.91E-01	4.91E-01	1.40E-01	1.44E-01	71.39	1.46E-01	1.51E-01	70.18	1.37E-01	1.41E-01	72.10	32	74
	2.00E-03	4.90E+00	4.90E+00	1.22E+00	1.22E+00	75.07	1.25E+00	1.25E+00	74.48	1.22E+00	1.22E+00	75.10	43	70

Table A-3. Performance results from NEA-SF-development with increased time step. Number of runs = 50, Field of view = 1.0^{o}.

Scenario: Northeast Asia to San Francisco with ascending orbit (NEA-SF-A)

This scenario utilizes two sensor satellites in a 1350 km altitude 58 degree inclination ascending node orbit. The target missile is an ICBM class missile launched from Northeast Asia targeting San Francisco. The missile flight time is 30 minutes. Figure A-19 is a diagram of the sensor satellites and target trajectories.

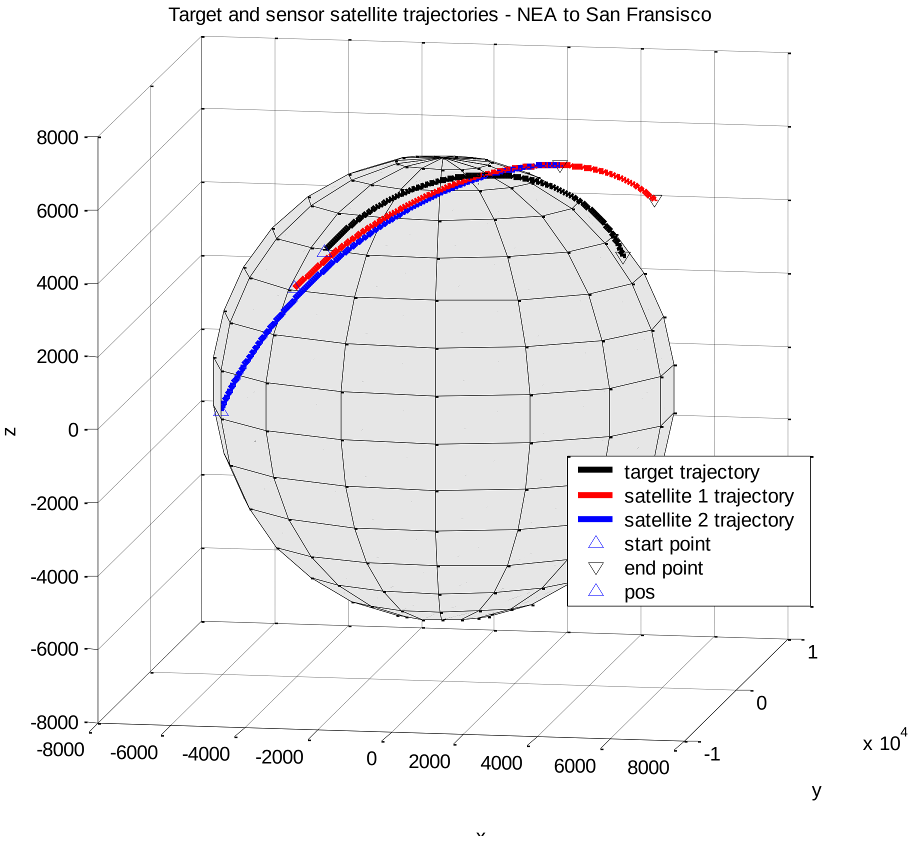


Figure A-19. Satellite and target trajectories for SEA-SF-A scenario

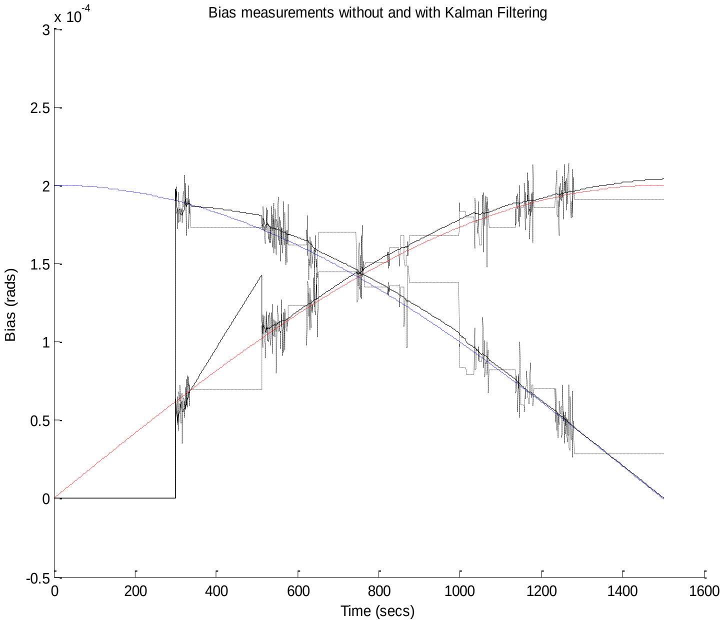


Figure A-20. Case 1: Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias

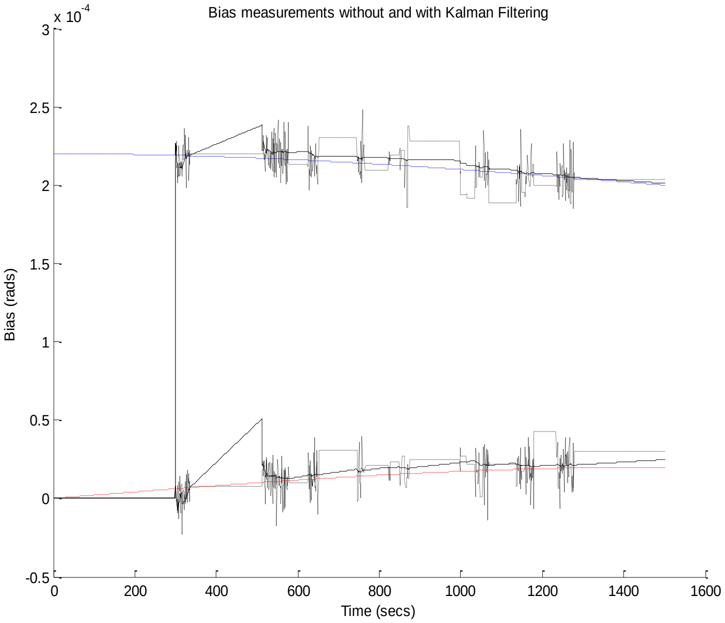


Figure A-21. Case 2: Sensor 1 bias measurements and estimates with and without bias EKF for constant bias with sinusoidal component.

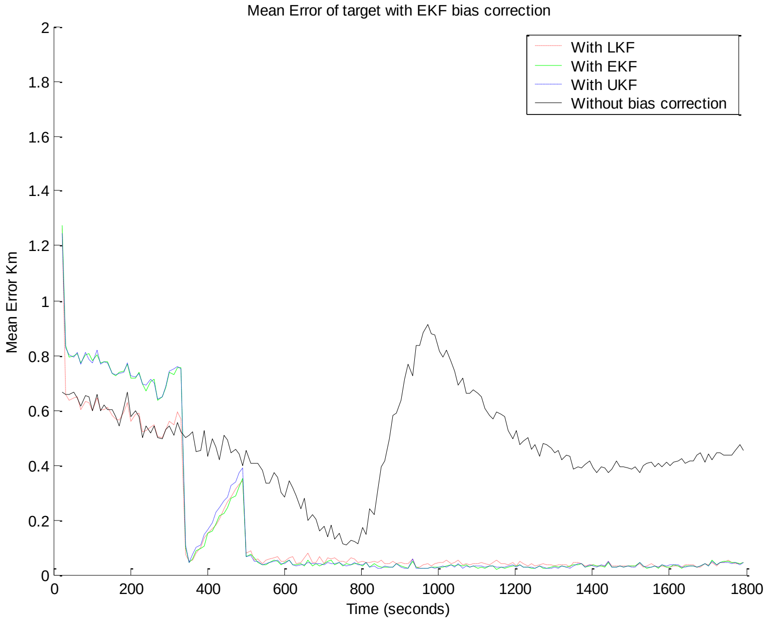


Figure A-22. Case 1: Mean error of target position for sinusoidal bias (β_0 = .2 *m*rads)

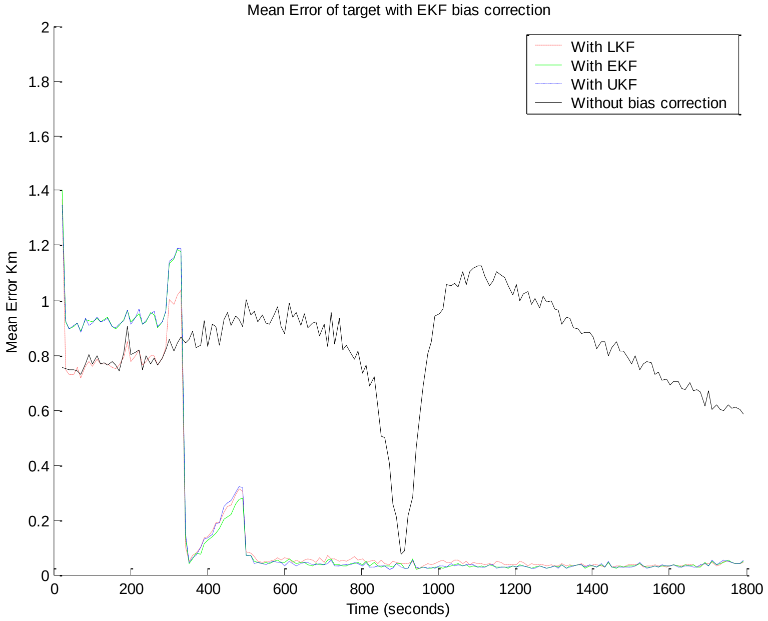


Figure A-23. Case 2: Mean error of target position for a constant bias with a sinusoidal component (β_0 = .2 *m*rads)

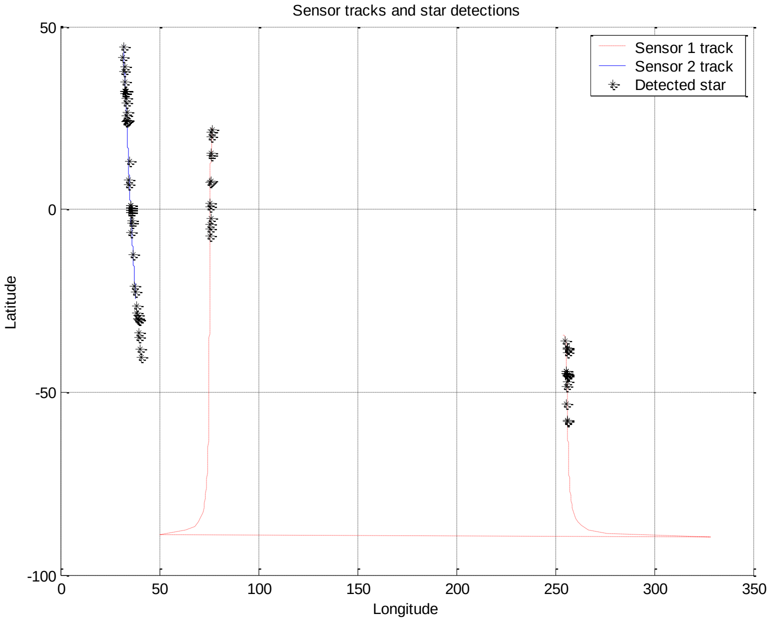


Figure A-24. Sensor trajectories with detected stars for NEA-SF-A scenario

NEA-SF-A Case 1														
Filter type		without bias correction		with EKF bias correction			with Sample and Hold correction			With EKF bias correction and post filtering			Numbe of stars	
	bias	Mean Err	STD	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Sensor 1	Sensor 2
LKF	0.00E+00	5.22E-02	8.33E-04	6.64E-02	1.48E-02	-27.22	5.97E-02	6.62E-03	-14.51	6.04E-02	1.55E-02	-15.77	313	359
	2.00E-05	6.86E-02	1.15E-03	6.79E-02	1.29E-02	1.07	6.10E-02	5.60E-03	11.10	6.21E-02	1.34E-02	9.51	314	361
	2.00E-04	4.51E-01	1.51E-03	7.04E-02	1.15E-02	84.41	7.46E-02	5.75E-03	83.48	6.49E-02	1.20E-02	85.61	316	360
	2.00E-03	4.50E+00	1.15E-03	1.30E-01	6.48E-03	97.12	3.89E-01	1.19E-02	91.35	1.30E-01	6.64E-03	97.12	280	353
EKF	0.00E+00	5.19E-02	1.03E-03	7.02E-02	1.39E-02	-35.28	5.82E-02	4.49E-03	-12.15	5.48E-02	1.55E-02	-5.66	312	361
	2.00E-05	6.87E-02	1.13E-03	6.34E-02	7.85E-03	7.71	6.00E-02	3.98E-03	12.66	4.88E-02	8.53E-03	28.98	314	359
	2.00E-04	4.51E-01	1.53E-03	7.39E-02	1.44E-02	83.62	7.49E-02	6.18E-03	83.39	6.19E-02	1.51E-02	86.27	314	361
	2.00E-03	4.50E+00	1.36E-03	1.34E-01	6.74E-03	97.02	3.85E-01	9.65E-03	91.44	1.54E-01	7.17E-03	96.59	279	350
UKF	0.00E+00	5.25E-02	8.88E-04	6.53E-02	1.19E-02	-24.29	5.97E-02	4.73E-03	-13.66	4.92E-02	1.24E-02	6.31	313	360
	2.00E-05	6.90E-02	1.50E-03	6.83E-02	1.25E-02	1.08	5.78E-02	5.46E-03	16.31	5.33E-02	1.42E-02	22.87	312	360
	2.00E-04	4.51E-01	1.59E-03	8.11E-02	1.64E-02	82.01	7.90E-02	1.02E-02	82.48	6.94E-02	1.77E-02	84.62	316	360
	2.00E-03	4.50E+00	1.39E-03	1.36E-01	6.96E-03	96.97	3.88E-01	1.34E-02	91.39	1.56E-01	7.09E-03	96.53	280	351
NEA-SF-A Case 2														
LKF	0.00E+00	5.22E-02	8.33E-04	6.64E-02	1.49E-02	-27.31	5.97E-02	6.62E-03	-14.51	6.05E-02	1.55E-02	-15.87	313	359
	2.00E-05	9.51E-02	1.61E-03	6.83E-02	1.29E-02	28.17	6.13E-02	6.07E-03	35.55	6.26E-02	1.34E-02	34.15	315	361
	2.00E-04	8.62E-01	1.69E-03	7.59E-02	1.16E-02	91.20	6.89E-02	3.43E-03	92.01	7.08E-02	1.21E-02	91.78	312	364
	2.00E-03	8.60E+00	1.44E-03	3.01E-01	5.41E-02	96.49	1.93E-01	3.83E-03	97.76	3.06E-01	5.38E-02	96.44	253	383
EKF	0.00E+00	5.19E-02	1.03E-03	7.02E-02	1.39E-02	-35.40	5.82E-02	4.49E-03	-12.15	5.49E-02	1.55E-02	-5.77	312	361
	2.00E-05	9.49E-02	1.65E-03	6.42E-02	7.93E-03	32.35	6.04E-02	3.89E-03	36.36	4.99E-02	8.73E-03	47.45	313	359
	2.00E-04	8.62E-01	1.75E-03	7.93E-02	1.43E-02	90.80	7.10E-02	4.63E-03	91.76	6.78E-02	1.51E-02	92.13	312	364
	2.00E-03	8.60E+00	1.37E-03	2.76E-01	6.41E-02	96.79	1.94E-01	3.85E-03	97.74	3.00E-01	5.93E-02	96.51	253	384
UKF	0.00E+00	5.25E-02	8.88E-04	6.53E-02	1.20E-02	-24.38	5.97E-02	4.73E-03	-13.66	4.92E-02	1.25E-02	6.21	313	360
	2.00E-05	9.52E-02	1.64E-03	6.87E-02	1.25E-02	27.83	5.83E-02	4.93E-03	38.79	5.38E-02	1.43E-02	43.47	313	360
	2.00E-04	8.62E-01	2.01E-03	8.64E-02	1.64E-02	89.97	7.20E-02	6.88E-03	91.64	7.50E-02	1.77E-02	91.30	311	363
	2.00E-03	8.60E+00	1.23E-03	2.90E-01	6.57E-02	96.62	1.94E-01	4.20E-03	97.75	3.15E-01	6.51E-02	96.34	253	383

Table A-4. Performance results from NEA-SF-A. Number of runs = 25, Field of view = 0.8^{o}, Start time 21 s, End time 1800 s

Scenario: Northeast Asia to San Francisco with descending orbit (NEA-SF-D)

This scenario utilizes two sensor satellites in a 1350 km altitude 58 degree inclination descending node orbit. The target missile is an ICBM class missile launched from Northeast Asia targeting San Francisco. The missile flight time is 30 minutes. Figure A-25 is a diagram of the sensor satellites and target trajectories.

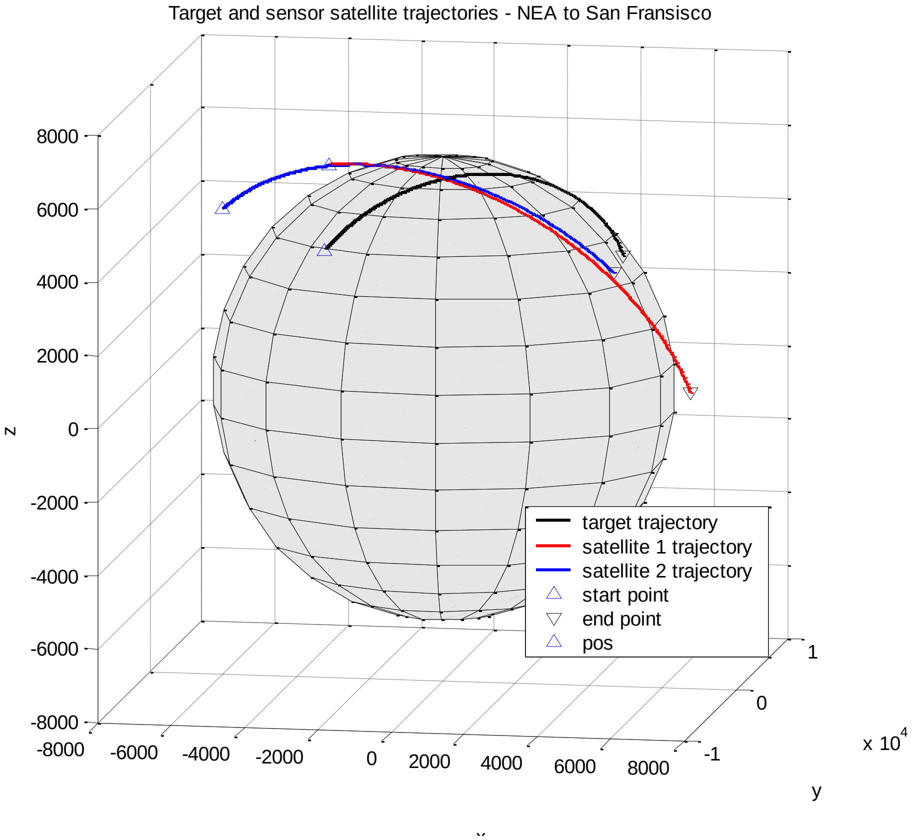


Figure A-25. Satellite and target trajectories for NEA-SF-D scenario

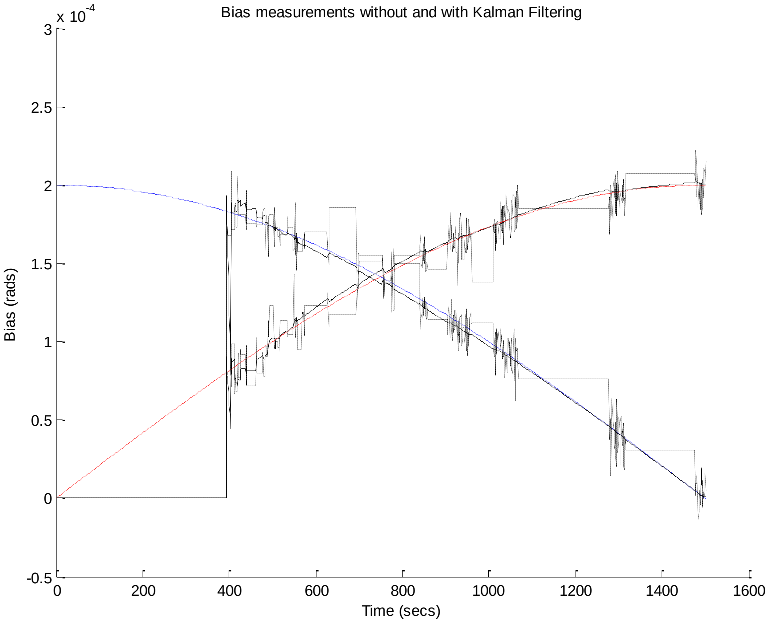


Figure A-26. Case 1: Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias

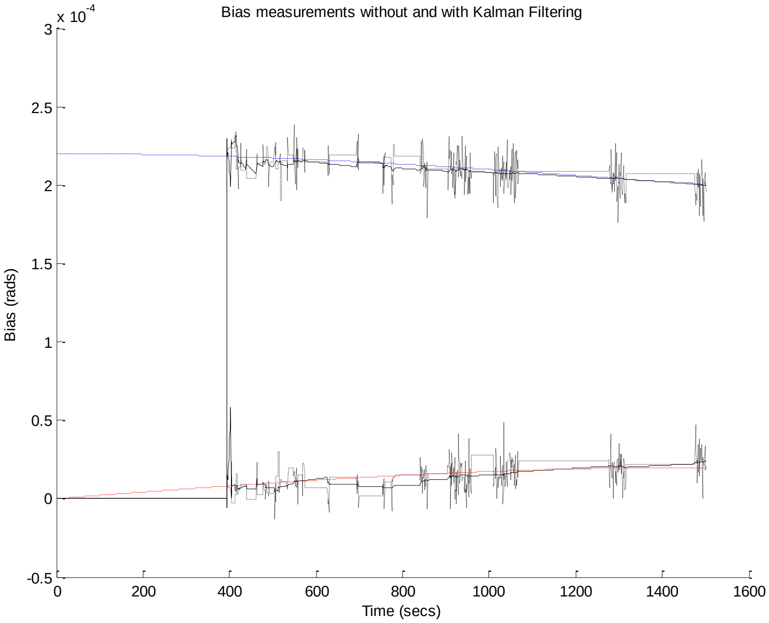
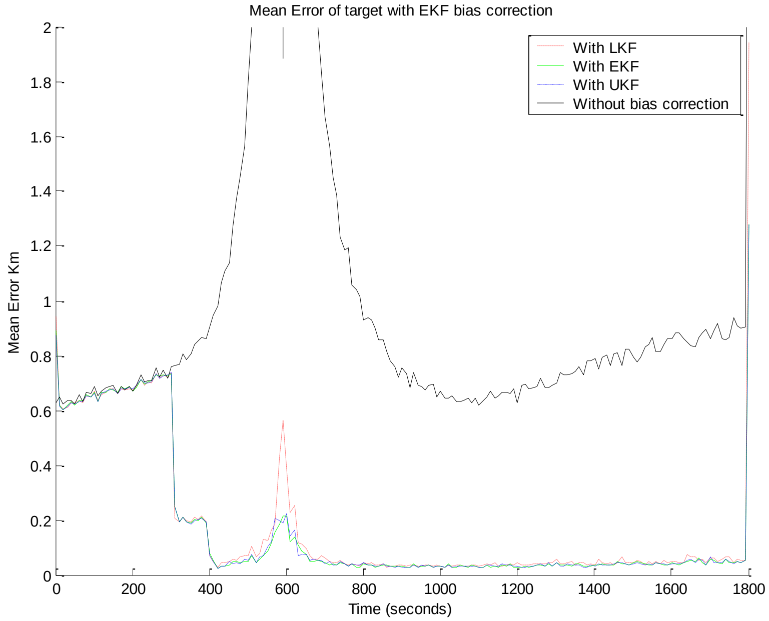


Figure A-27. Case 2: Sensor 1 bias measurements and estimates with and without bias EKF for constant bias with sinusoidal component.



.

Figure A-28. Case 1: Mean error of target position for sinusoidal bias (β_0 = .2 *m*rads)

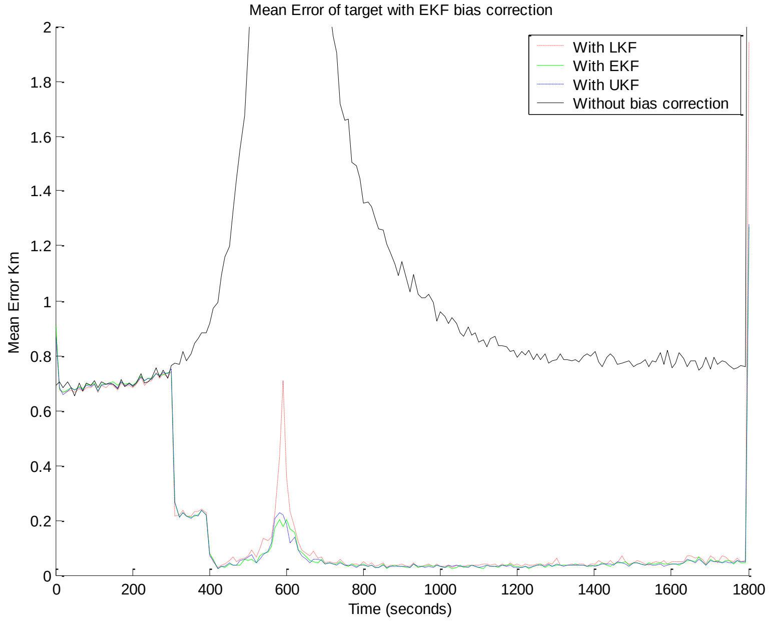


Figure A-29. Case 2: Bias measurements and estimates with and without bias EKF for constant bias with sinusoidal component.

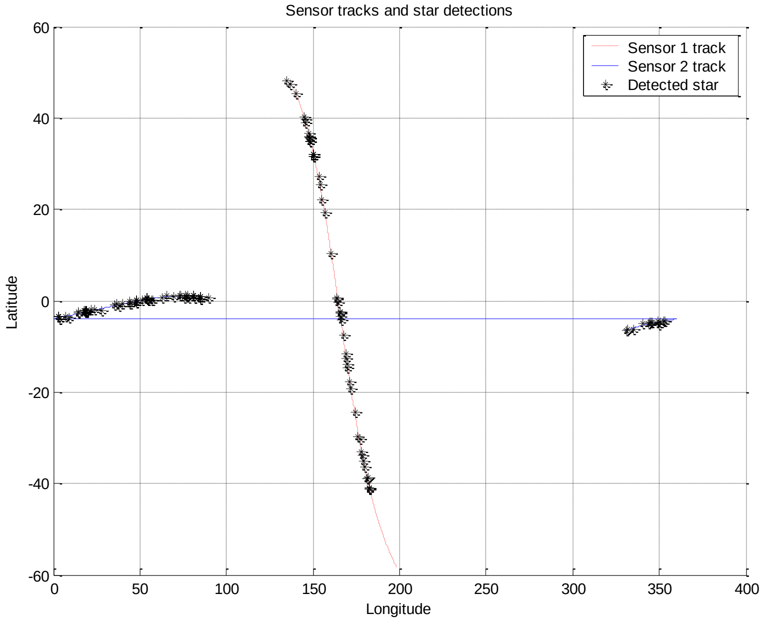


Figure A-30. Sensor trajectories with detected stars for NEA-SF-D scenario

NEA-SF-D Case 1														
Filter type		without bias correction		with EKF bias correction			with Sample and Hold correction			With EKF bias correction and post filtering			Numbe of stars	
	bias	Mean Err	STD	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Sensor 1	Sensor 2
LKF	0.00E+00	7.91E-02	2.58E-03	8.01E-02	3.38E-03	-1.28	8.29E-02	5.33E-03	-4.73	6.06E-02	2.80E-03	23.43	408	429
	2.00E-05	1.53E-01	3.14E-03	8.17E-02	3.49E-03	46.53	8.60E-02	7.75E-03	43.69	6.16E-02	3.13E-03	59.68	408	429
	2.00E-04	1.38E+00	2.84E-03	9.86E-02	1.91E-03	92.84	1.12E-01	5.77E-03	91.90	7.93E-02	1.95E-03	94.24	409	428
	2.00E-03	1.36E+01	3.43E-03	2.98E-01	2.61E-03	97.81	5.52E-01	1.18E-02	95.94	2.84E-01	3.15E-03	97.92	402	421
EKF	0.00E+00	7.97E-02	2.76E-03	8.06E-02	2.77E-03	-1.16	8.63E-02	6.25E-03	-8.27	4.60E-02	2.61E-03	42.31	408	427
	2.00E-05	1.51E-01	3.27E-03	8.06E-02	2.89E-03	46.68	8.38E-02	4.26E-03	44.61	4.48E-02	3.20E-03	70.36	407	428
	2.00E-04	1.38E+00	3.14E-03	9.85E-02	2.60E-03	92.85	1.11E-01	6.52E-03	91.94	6.34E-02	2.73E-03	95.40	410	425
	2.00E-03	1.36E+01	4.52E-03	2.97E-01	2.30E-03	97.82	5.58E-01	1.19E-02	95.90	2.68E-01	3.24E-03	98.03	402	421
UKF	0.00E+00	7.94E-02	2.33E-03	8.04E-02	2.55E-03	-1.25	8.23E-02	3.50E-03	-3.57	4.50E-02	2.97E-03	43.41	407	429
	2.00E-05	1.51E-01	3.56E-03	8.10E-02	2.66E-03	46.31	8.54E-02	4.42E-03	43.44	4.56E-02	3.15E-03	69.77	407	428
	2.00E-04	1.38E+00	4.60E-03	9.80E-02	2.65E-03	92.88	1.12E-01	8.13E-03	91.86	6.36E-02	2.06E-03	95.38	409	425
	2.00E-03	1.36E+01	4.80E-03	2.97E-01	3.59E-03	97.82	5.51E-01	8.46E-03	95.95	2.69E-01	4.24E-03	98.03	402	420
NEA-SF-D Case 2														
LKF	0.00E+00	7.98E-02	2.54E-03	8.05E-02	2.60E-03	-0.92	8.53E-02	7.37E-03	-6.86	6.03E-02	2.37E-03	24.46	407	428
	2.00E-05	1.79E-01	4.42E-03	8.10E-02	2.60E-03	54.84	8.66E-02	5.98E-03	51.75	6.10E-02	2.55E-03	66.00	406	427
	2.00E-04	1.69E+00	4.07E-03	9.90E-02	2.69E-03	94.16	1.03E-01	3.99E-03	93.94	8.03E-02	2.89E-03	95.26	407	427
	2.00E-03	1.69E+01	3.58E-03	3.18E-01	2.39E-03	98.12	3.17E-01	6.13E-03	98.13	3.07E-01	2.59E-03	98.19	421	437
EKF	0.00E+00	7.92E-02	2.46E-03	8.05E-02	2.85E-03	-1.73	8.45E-02	5.85E-03	-6.74	4.60E-02	4.34E-03	41.87	408	428
	2.00E-05	1.80E-01	2.55E-03	8.09E-02	2.33E-03	55.18	8.37E-02	3.41E-03	53.61	4.52E-02	3.01E-03	74.93	407	428
	2.00E-04	1.70E+00	4.43E-03	1.00E-01	2.38E-03	94.11	1.04E-01	5.47E-03	93.89	6.55E-02	3.19E-03	96.14	407	428
	2.00E-03	1.69E+01	4.14E-03	3.20E-01	2.96E-03	98.11	3.17E-01	7.14E-03	98.12	2.92E-01	3.53E-03	98.27	422	432
UKF	0.00E+00	7.94E-02	2.44E-03	8.04E-02	2.50E-03	-1.33	8.25E-02	5.28E-03	-3.92	4.56E-02	2.36E-03	42.49	408	430
	2.00E-05	1.79E-01	3.30E-03	8.17E-02	2.48E-03	54.41	8.41E-02	4.98E-03	53.03	4.65E-02	2.57E-03	74.02	409	428
	2.00E-04	1.70E+00	4.53E-03	9.96E-02	3.18E-03	94.13	1.02E-01	4.70E-03	93.99	6.57E-02	3.20E-03	96.12	408	428
	2.00E-03	1.69E+01	3.37E-03	3.21E-01	4.14E-03	98.10	3.21E-01	6.42E-03	98.11	2.95E-01	5.08E-03	98.25	422	431

Table A-5: Performance results from NEA-SF-D. Number of runs = 25, Field of view = 0.8^{o}, Start time 0 s, End time 1800 s

Scenario: Northeast Asia to Hawaii with ascending orbit (NEA-HI-A)

This scenario utilizes two sensor satellites in a 1350 km altitude 58 degree inclination ascending node orbit. The target missile is an ICBM class missile launched from Northeast Asia targeting Hawaii. The missile flight time is 25 minutes. Figure A-31 is a diagram of the sensor satellites and target trajectories.

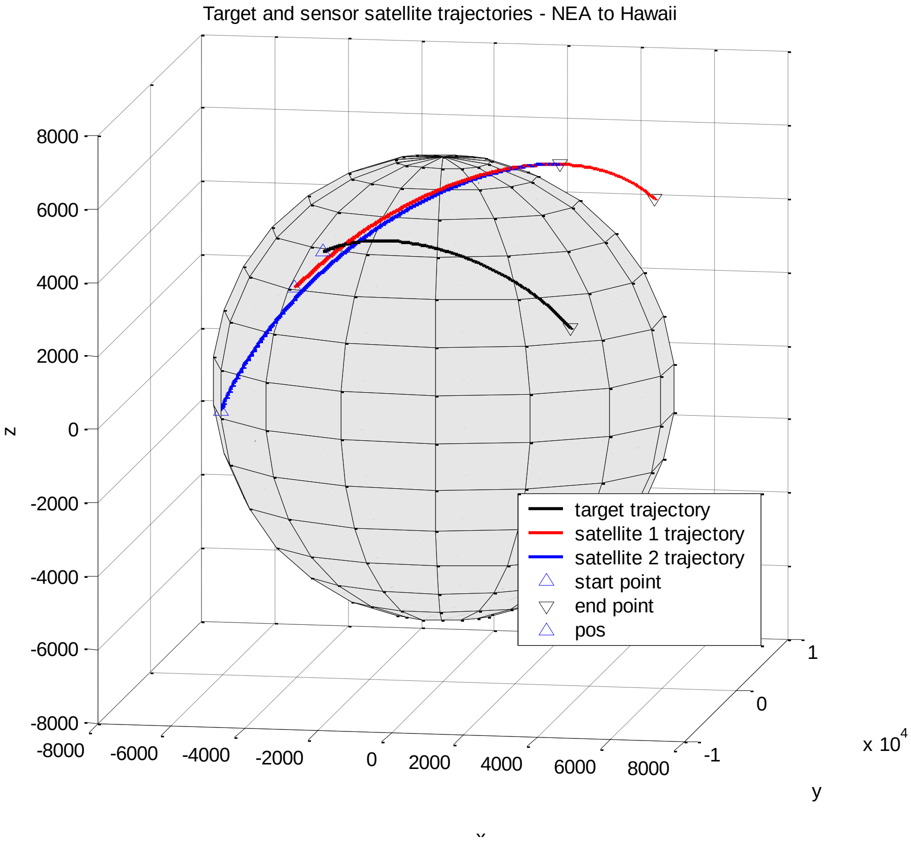


Figure A-31. Satellite and target trajectories for NEA-HI-A scenario

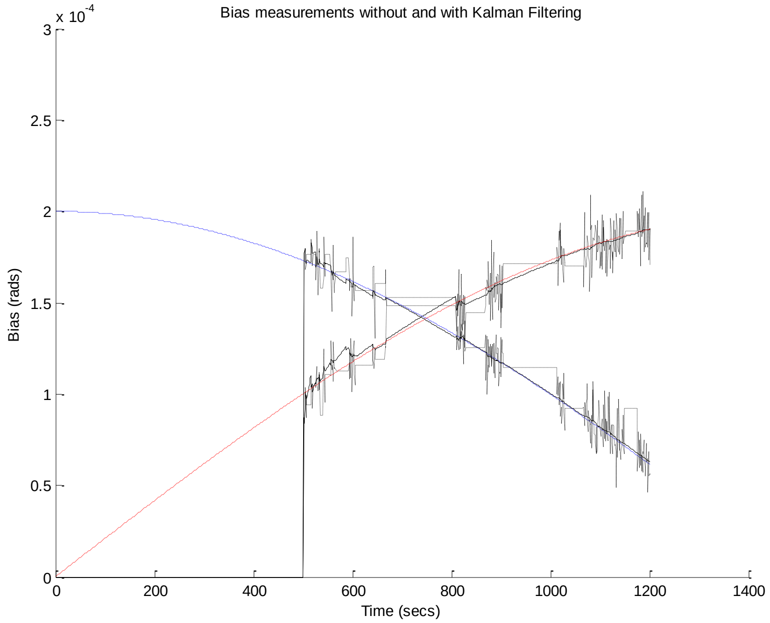


Figure A-32. Case 1: Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias

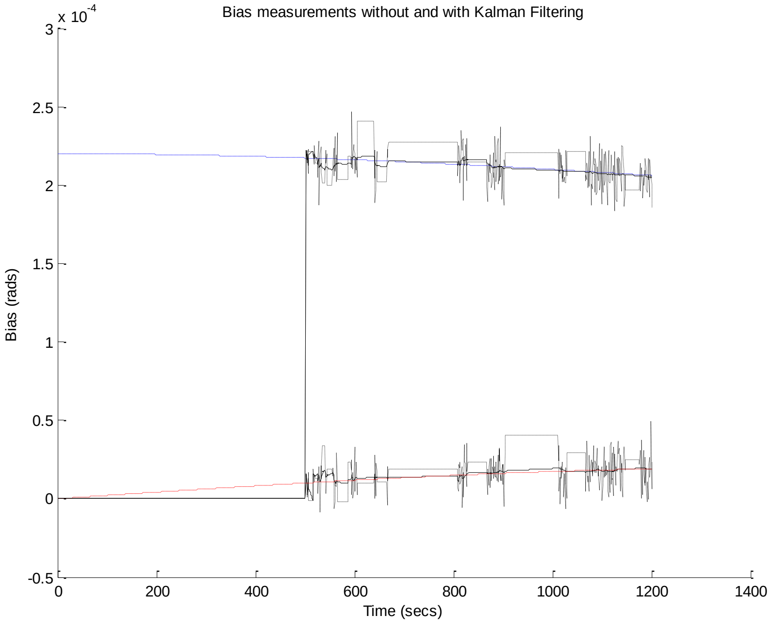


Figure A-33. Case 2: Sensor 1 bias measurements and estimates with and without bias EKF for constant bias with sinusoidal component.

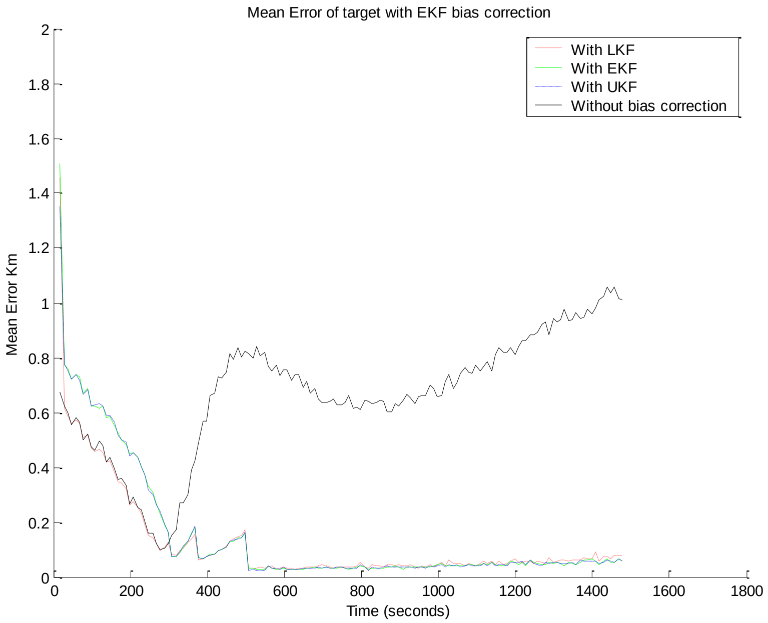


Figure A-34. Case 1: Mean error of target position for sinusoidal bias (β_0 = .2 *m*rads)

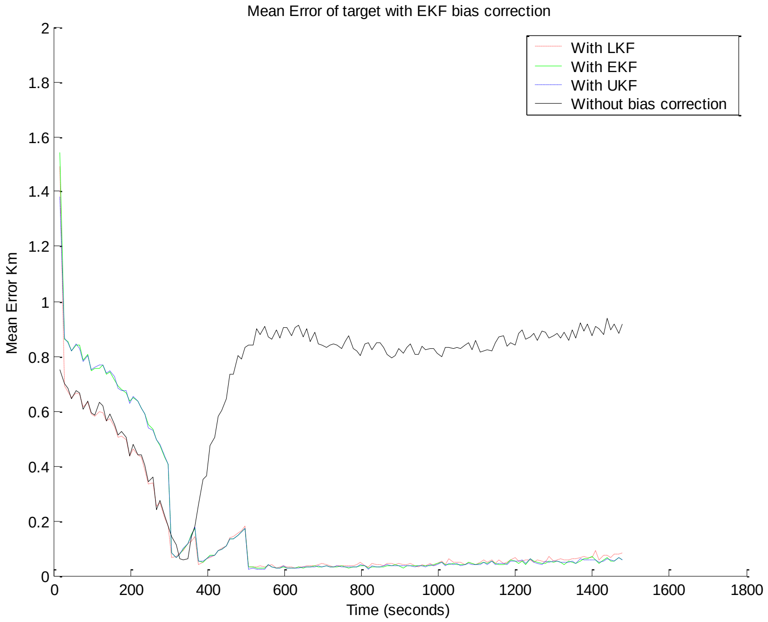


Figure A-35. Case 2: Bias measurements and estimates with and without bias EKF for constant bias with sinusoidal component.

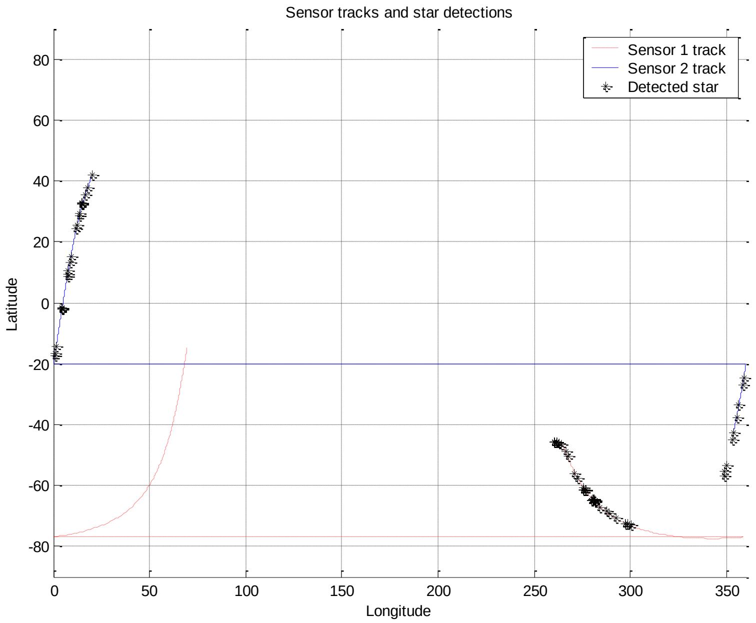


Figure A-36. Sensor trajectories with detected stars for NEA-HI-A scenario

NEA-HI-A Case 1														
Filter type		without bias correction		with EKF bias correction			with Sample and Hold correction			With EKF bias correction and post filtering			Numbe of stars	
	bias	Mean Err	STD	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Sensor 1	Sensor 2
LKF	0.00E+00	4.84E-02	9.59E-04	5.45E-02	5.08E-03	-12.75	5.32E-02	2.72E-03	-9.96	4.48E-02	5.22E-03	7.43	460	212
	2.00E-05	8.08E-02	7.45E-04	5.30E-02	3.24E-03	34.49	5.43E-02	3.38E-03	32.84	4.30E-02	3.42E-03	46.83	461	212
	2.00E-04	6.84E-01	1.23E-03	6.43E-02	2.99E-03	90.60	7.04E-02	3.53E-03	89.72	5.48E-02	3.17E-03	91.99	458	216
	2.00E-03	6.85E+00	1.33E-03	2.28E-01	2.04E-03	96.67	4.34E-01	6.35E-03	93.66	2.22E-01	2.56E-03	96.76	426	214
EKF	0.00E+00	4.86E-02	6.72E-04	5.14E-02	2.32E-03	-5.61	5.35E-02	2.41E-03	-10.04	3.75E-02	3.67E-03	22.84	462	212
	2.00E-05	8.02E-02	1.19E-03	5.40E-02	4.33E-03	32.65	5.38E-02	4.00E-03	32.89	4.02E-02	5.58E-03	49.92	461	212
	2.00E-04	6.85E-01	1.43E-03	6.53E-02	3.50E-03	90.46	7.10E-02	3.84E-03	89.63	5.14E-02	3.64E-03	92.50	459	214
	2.00E-03	6.85E+00	8.74E-04	2.28E-01	1.53E-03	96.68	4.34E-01	7.57E-03	93.67	2.18E-01	1.47E-03	96.82	468	215
UKF	0.00E+00	4.83E-02	7.43E-04	5.34E-02	4.55E-03	-10.63	5.25E-02	2.84E-03	-8.74	3.90E-02	4.42E-03	19.31	461	212
	2.00E-05	8.08E-02	8.56E-04	5.55E-02	5.67E-03	31.34	5.35E-02	3.19E-03	33.78	4.15E-02	6.49E-03	48.66	462	212
	2.00E-04	6.85E-01	8.29E-04	6.54E-02	3.47E-03	90.46	7.09E-02	3.74E-03	89.65	5.15E-02	3.54E-03	92.47	457	215
	2.00E-03	6.85E+00	9.99E-04	2.28E-01	1.73E-03	96.67	4.34E-01	7.64E-03	93.66	2.18E-01	2.31E-03	96.82	468	214
NEA-HI-A Case 2														
LKF	0.00E+00	4.85E-02	7.32E-04	5.51E-02	5.20E-03	-13.64	5.31E-02	4.34E-03	-9.58	4.56E-02	5.33E-03	5.98	460	212
	2.00E-05	8.77E-02	7.88E-04	5.43E-02	5.56E-03	38.05	5.28E-02	3.46E-03	39.84	4.48E-02	5.53E-03	48.94	461	213
	2.00E-04	7.64E-01	9.20E-04	6.61E-02	5.22E-03	91.35	6.39E-02	2.78E-03	91.63	5.64E-02	5.47E-03	92.62	458	217
	2.00E-03	7.64E+00	8.84E-04	2.14E-01	2.48E-03	97.20	2.20E-01	5.73E-03	97.12	2.08E-01	3.13E-03	97.27	434	216
EKF	0.00E+00	4.83E-02	1.08E-03	5.39E-02	4.34E-03	-11.45	5.39E-02	3.11E-03	-11.45	3.99E-02	4.93E-03	17.48	462	213
	2.00E-05	8.81E-02	1.07E-03	5.42E-02	3.89E-03	38.41	5.28E-02	3.01E-03	40.00	4.01E-02	4.59E-03	54.42	460	213
	2.00E-04	7.64E-01	7.95E-04	6.44E-02	3.57E-03	91.57	6.56E-02	3.44E-03	91.42	5.04E-02	4.29E-03	93.41	460	217
	2.00E-03	7.64E+00	1.08E-03	2.13E-01	1.74E-03	97.21	2.19E-01	3.87E-03	97.13	2.05E-01	2.26E-03	97.32	434	216
UKF	0.00E+00	4.83E-02	6.90E-04	5.44E-02	6.01E-03	-12.63	5.18E-02	3.13E-03	-7.30	4.06E-02	6.63E-03	15.98	458	212
	2.00E-05	8.83E-02	1.06E-03	5.42E-02	3.70E-03	38.67	5.36E-02	2.26E-03	39.27	3.98E-02	3.42E-03	54.92	460	213
	2.00E-04	7.65E-01	9.99E-04	6.34E-02	3.13E-03	91.70	6.43E-02	3.63E-03	91.59	4.92E-02	3.05E-03	93.56	458	218
	2.00E-03	7.64E+00	9.99E-04	2.14E-01	2.81E-03	97.20	2.20E-01	4.06E-03	97.12	2.06E-01	2.75E-03	97.31	435	216

Table A-6. Performance results from NEA-HI-A. Number of runs = 25, Field of view = 0.8^{o}, Start time 17 s, End time 1482 s

Scenario: Northeast Asia to Hawaii with descending orbit (NEA-HI-D)

This scenario utilizes two sensor satellites in a 1350 km altitude 58 degree inclination descending node orbit. The target missile is an ICBM class missile launched from Northeast Asia targeting Hawaii. The missile flight time is 30 minutes. Figure A-37 is a diagram of the sensor satellites and target trajectories.

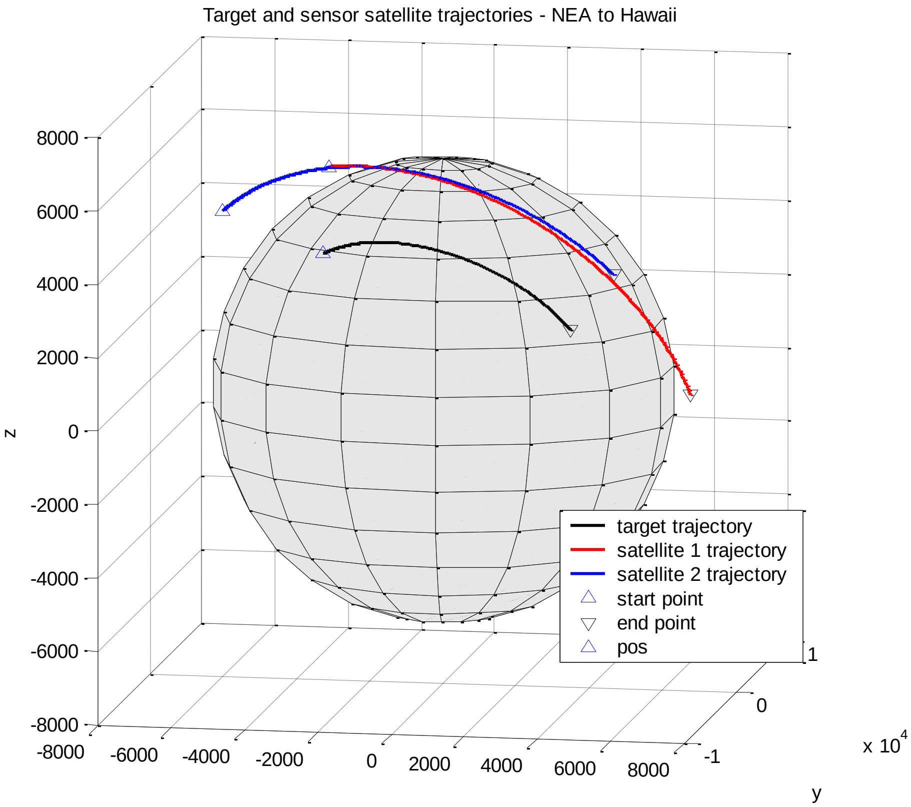


Figure A-37. Satellite and target for NEA-HI-D scenario

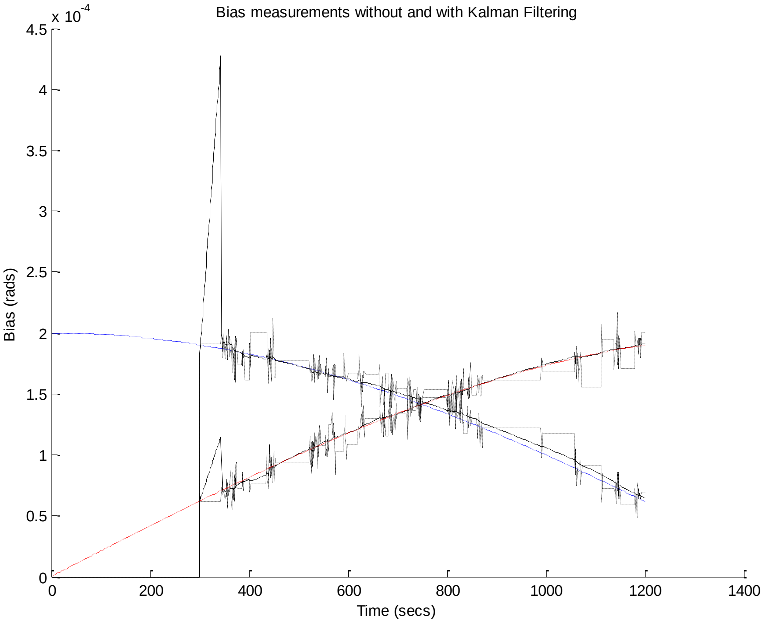


Figure A-38. Case 1: Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias

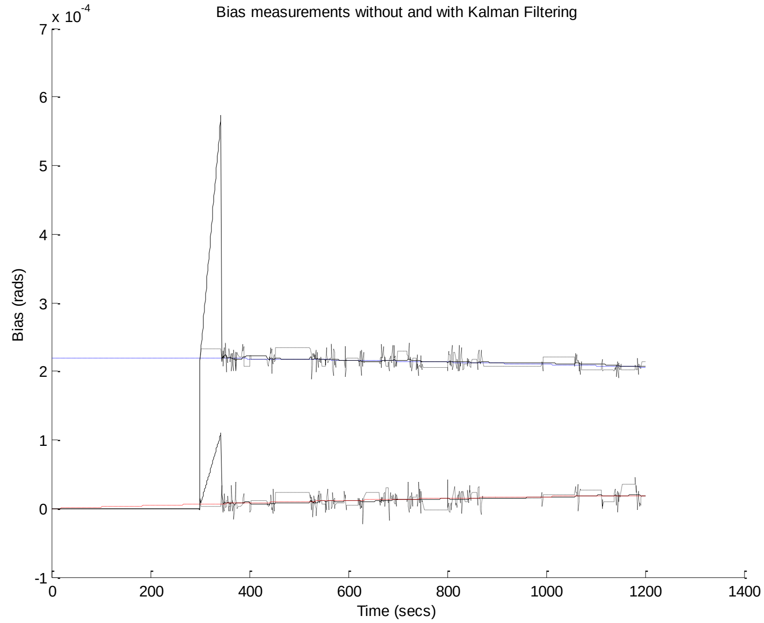


Figure A-39. Case 2: Sensor 1 bias measurements and estimates with and without bias EKF for constant bias with sinusoidal component.

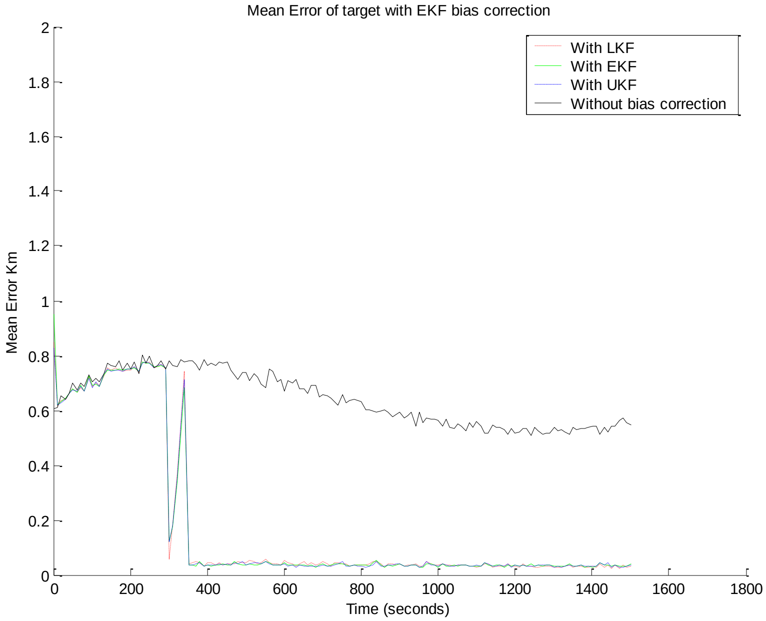


Figure A-40. Case 1: Mean error of target position for sinusoidal bias (β_0 = .2 *m*rads)

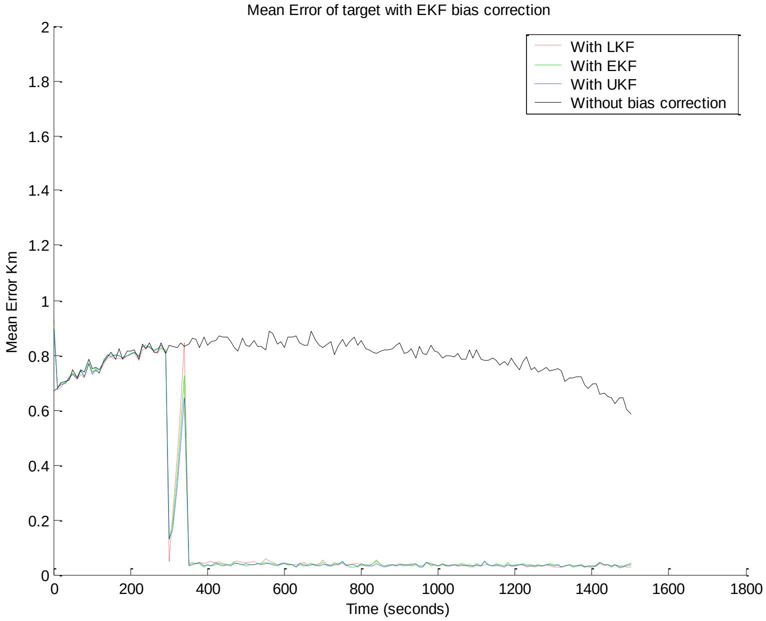


Figure A-41. Case 2: Bias measurements and estimates with and without bias EKF for constant bias with sinusoidal component.

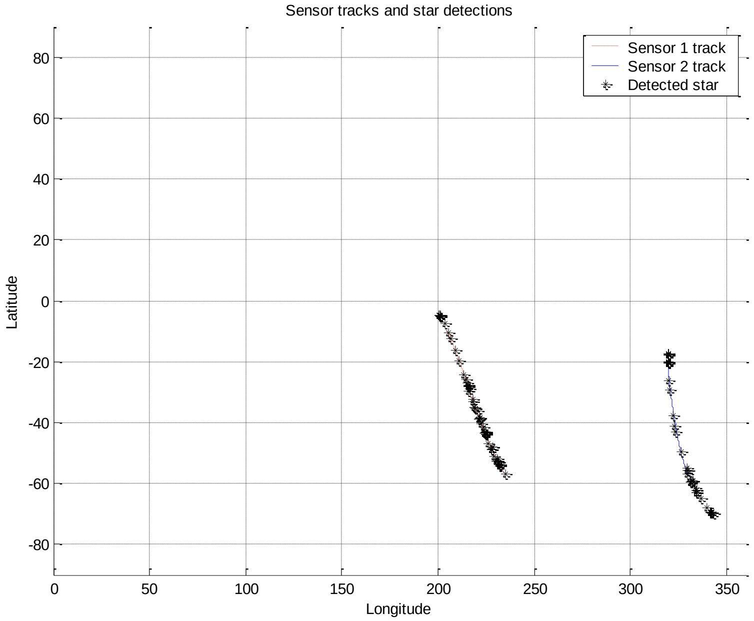


Figure A-42. Sensor trajectories with detected stars for NEA-HI-D scenario

NEA-HI-D Case 1															
Filter type		without bias correction		with EKF bias correction			with Sample and Hold correction			With EKF bias correction and post filtering			Numbe of stars	
	bias	Mean Err	STD	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Sensor 1	Sensor 2
LKF	0.00E+00	4.73E-02	8.54E-04	6.54E-02	9.36E-03	-38.16	5.41E-02	3.73E-03	-14.26	5.67E-02	9.50E-03	-19.76	396	281
	2.00E-05	7.80E-02	1.01E-03	6.65E-02	9.00E-03	14.68	5.40E-02	4.04E-03	30.79	5.77E-02	9.45E-03	25.93	396	282
	2.00E-04	6.48E-01	1.24E-03	6.58E-02	8.90E-03	89.85	6.12E-02	5.07E-03	90.55	5.76E-02	9.21E-03	91.11	395	284
	2.00E-03	6.47E+00	1.01E-03	9.04E-02	2.12E-02	98.60	2.55E-01	8.93E-03	96.06	8.99E-02	2.20E-02	98.61	380	285
EKF	0.00E+00	4.75E-02	6.08E-04	6.47E-02	7.09E-03	-36.26	5.37E-02	3.31E-03	-13.11	5.01E-02	6.23E-03	-5.45	395	281
	2.00E-05	7.79E-02	8.04E-04	6.48E-02	7.95E-03	16.76	5.39E-02	3.76E-03	30.84	5.04E-02	7.79E-03	35.27	397	282
	2.00E-04	6.47E-01	8.16E-04	6.93E-02	8.74E-03	89.30	6.20E-02	5.07E-03	90.42	5.46E-02	7.27E-03	91.56	396	282
	2.00E-03	6.46E+00	1.09E-03	8.67E-02	1.69E-02	98.66	2.56E-01	1.12E-02	96.04	8.27E-02	1.49E-02	98.72	382	281
UKF	0.00E+00	4.75E-02	6.69E-04	6.56E-02	1.20E-02	-38.03	5.45E-02	4.27E-03	-14.74	5.03E-02	1.14E-02	-5.75	397	281
	2.00E-05	7.81E-02	7.35E-04	6.67E-02	8.66E-03	14.61	5.50E-02	4.45E-03	29.62	5.22E-02	7.82E-03	33.17	396	282
	2.00E-04	6.47E-01	1.04E-03	6.66E-02	1.19E-02	89.71	6.01E-02	4.14E-03	90.72	5.24E-02	1.01E-02	91.91	397	283
	2.00E-03	6.47E+00	8.89E-04	8.53E-02	1.83E-02	98.68	2.54E-01	8.72E-03	96.07	8.14E-02	1.64E-02	98.74	380	285
NEA-HI-D Case 2															
LKF	0.00E+00	4.72E-02	6.89E-04	6.68E-02	1.14E-02	-41.48	5.52E-02	3.55E-03	-16.93	5.78E-02	1.21E-02	-22.53	396	281
	2.00E-05	9.27E-02	8.84E-04	6.22E-02	9.03E-03	32.93	5.39E-02	3.47E-03	41.85	5.32E-02	9.50E-03	42.64	395	282
	2.00E-04	8.29E-01	1.05E-03	6.67E-02	9.25E-03	91.95	5.44E-02	3.15E-03	93.44	5.83E-02	9.61E-03	92.96	400	283
	2.00E-03	8.28E+00	1.06E-03	2.25E+00	9.77E-03	72.77	5.73E-02	3.63E-03	99.31	2.32E+00	9.85E-03	71.92	406	296
EKF	0.00E+00	4.73E-02	6.09E-04	6.38E-02	1.06E-02	-34.93	5.51E-02	4.37E-03	-16.66	4.90E-02	8.98E-03	-3.66	397	281
	2.00E-05	9.26E-02	8.38E-04	6.41E-02	8.83E-03	30.79	5.47E-02	4.45E-03	40.94	4.95E-02	7.96E-03	46.61	395	282
	2.00E-04	8.29E-01	8.43E-04	6.75E-02	8.56E-03	91.86	5.37E-02	3.62E-03	93.52	5.41E-02	8.49E-03	93.47	400	284
	2.00E-03	8.28E+00	9.60E-04	2.25E+00	1.07E-02	72.79	5.63E-02	4.47E-03	99.32	2.32E+00	1.11E-02	71.98	406	296
UKF	0.00E+00	4.74E-02	6.48E-04	6.73E-02	1.01E-02	-41.93	5.47E-02	3.87E-03	-15.26	5.26E-02	1.00E-02	-10.82	397	282
	2.00E-05	9.23E-02	7.62E-04	6.30E-02	9.04E-03	31.75	5.52E-02	2.81E-03	40.21	4.83E-02	9.06E-03	47.71	397	282
	2.00E-04	8.28E-01	1.25E-03	6.62E-02	1.10E-02	92.01	5.52E-02	5.09E-03	93.34	5.32E-02	1.05E-02	93.58	402	283
	2.00E-03	8.28E+00	8.80E-04	2.25E+00	1.11E-02	72.78	5.86E-02	4.00E-03	99.29	2.32E+00	1.14E-02	71.97	408	296

Table A-7. Performance results from NEA-HI-D. Number of runs = 25, Field of view = 0.8^o, Start time 0 s, End time 1500 s

Scenario: Northeast Asia to Guam with ascending orbit (NEA-GU-A)

This scenario utilizes two sensor satellites in a 1350 km altitude 58 degree inclination ascending node orbit. The target missile is an ICBM class missile launched from Northeast Asia targeting Guam. The missile flight time is 20 minutes. Figure A-43 is a diagram of the sensor satellites and target trajectories.

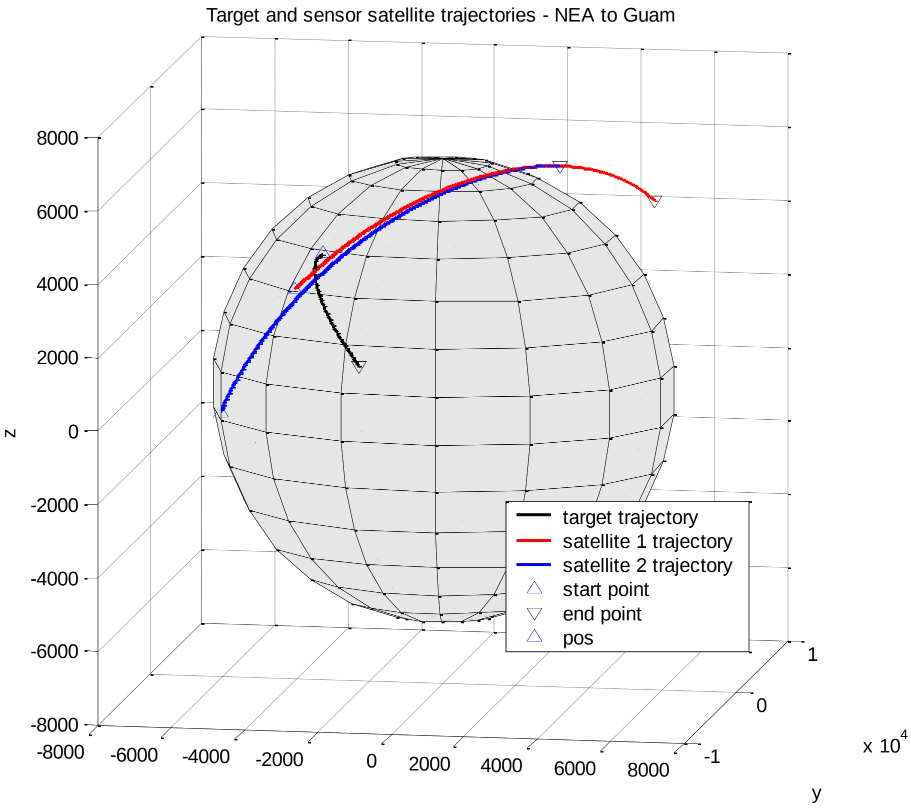


Figure A-43. Satellite and target for NEA-GU-A scenario

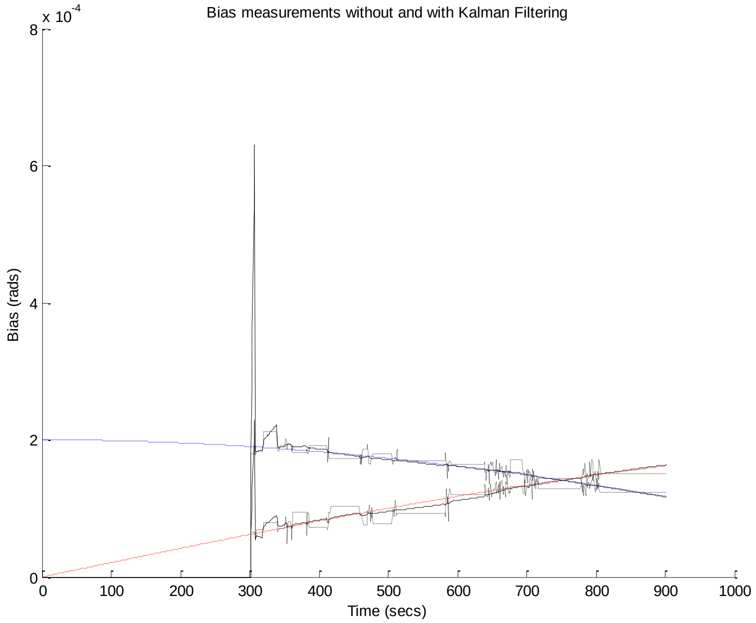


Figure A-44. Case 1: Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias

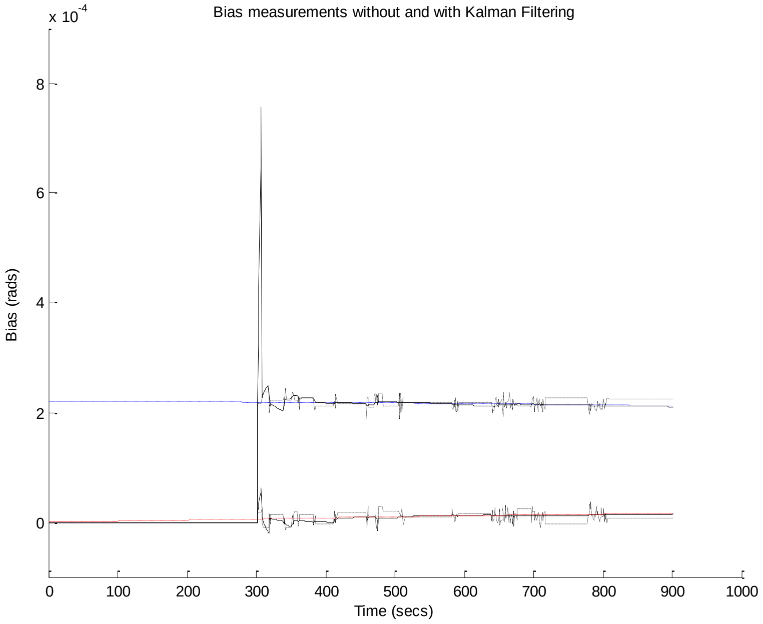


Figure A-45. Case 2: Sensor 1 bias measurements and estimates with and without bias EKF for constant bias with sinusoidal component.

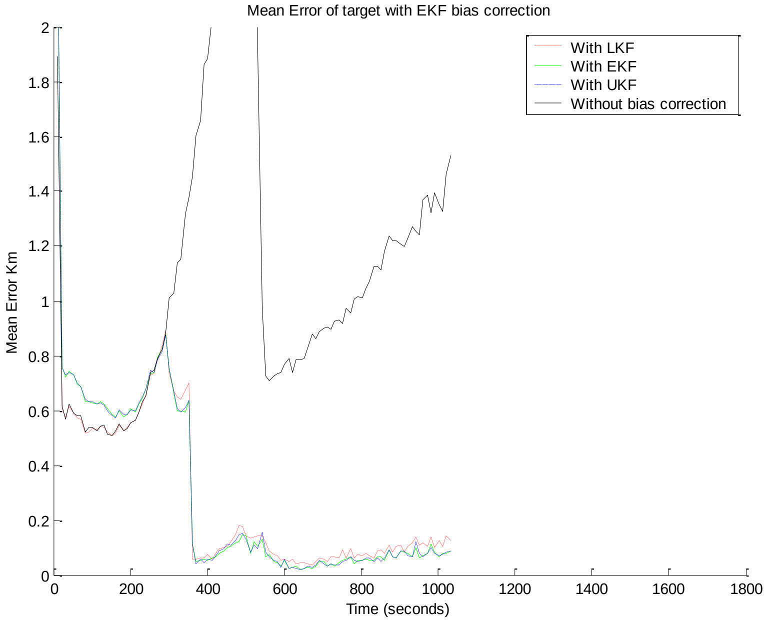


Figure A-46. Case 1: Mean error of target position for sinusoidal bias (β_0 = .2 *m*rads)

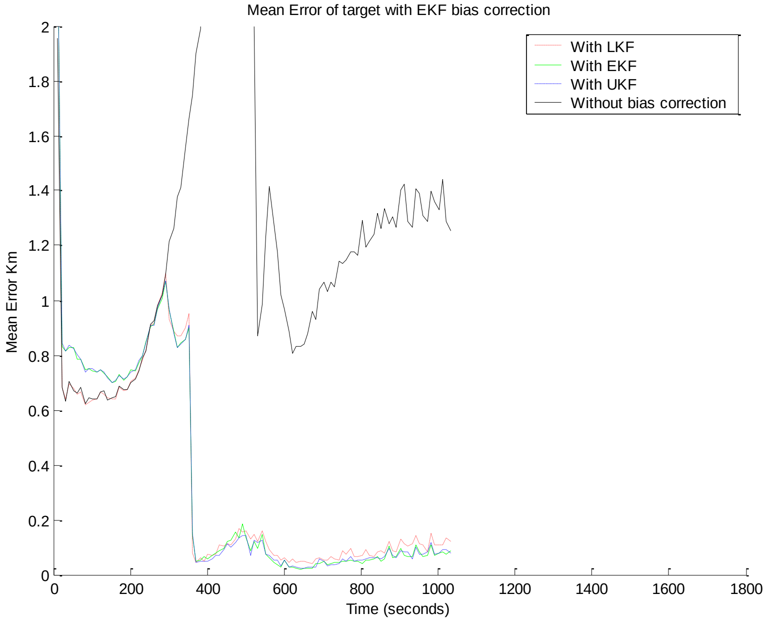


Figure A-47. Case 2: Bias measurements and estimates with and without bias EKF for constant bias with sinusoidal component.

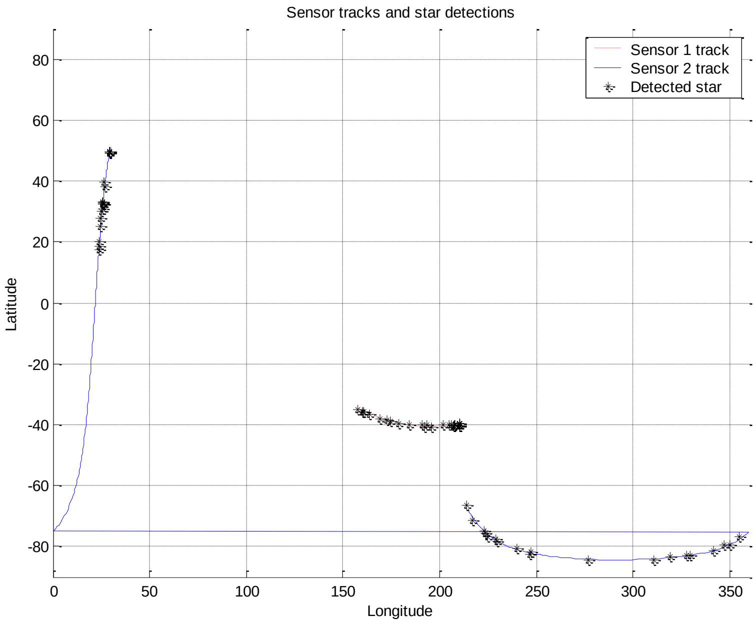


Figure A-48. Sensor trajectories with detected stars for NEA-GU-A scenario

NEA-GU-A Case 1														
Filter type		without bias correction		with EKF bias correction			with Sample and Hold correction			With EKF bias correction and post filtering			Numbe of stars	
	bias	Mean Err	STD	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Sensor 1	Sensor 2
LKF	0.00E+00	1.01E-01	2.47E-03	1.05E-01	5.23E-03	-4.03	1.17E-01	6.17E-03	-15.64	7.83E-02	6.61E-03	22.71	174	157
	2.00E-05	1.70E-01	3.10E-03	1.11E-01	7.53E-03	34.46	1.23E-01	9.83E-03	27.58	8.50E-02	9.62E-03	49.90	174	156
	2.00E-04	1.46E+00	2.55E-03	1.56E-01	7.03E-03	89.33	1.74E-01	1.36E-02	88.03	1.32E-01	8.24E-03	90.96	176	151
	2.00E-03	1.45E+01	3.66E-03	5.71E-01	5.58E-03	96.07	9.34E-01	2.49E-02	93.58	5.63E-01	6.39E-03	96.13	186	151
EKF	0.00E+00	1.02E-01	2.65E-03	1.07E-01	6.32E-03	-5.11	1.19E-01	1.22E-02	-16.62	6.13E-02	8.21E-03	39.71	176	157
	2.00E-05	1.70E-01	3.96E-03	1.08E-01	5.32E-03	36.64	1.19E-01	1.11E-02	30.08	6.42E-02	7.79E-03	62.18	176	156
	2.00E-04	1.46E+00	4.08E-03	1.55E-01	7.10E-03	89.40	1.74E-01	1.40E-02	88.09	1.10E-01	9.22E-03	92.49	175	151
	2.00E-03	1.45E+01	3.61E-03	5.70E-01	5.79E-03	96.08	9.24E-01	2.84E-02	93.65	5.13E-01	7.46E-03	96.47	185	151
UKF	0.00E+00	1.01E-01	2.99E-03	1.07E-01	5.72E-03	-5.87	1.14E-01	9.89E-03	-12.90	6.27E-02	8.80E-03	37.87	175	157
	2.00E-05	1.70E-01	3.85E-03	1.08E-01	5.27E-03	36.55	1.26E-01	1.58E-02	25.73	6.44E-02	9.18E-03	62.10	175	156
	2.00E-04	1.46E+00	2.47E-03	1.53E-01	6.51E-03	89.54	1.75E-01	1.32E-02	88.01	1.07E-01	9.65E-03	92.65	175	151
	2.00E-03	1.45E+01	3.77E-03	5.71E-01	6.32E-03	96.07	9.34E-01	3.17E-02	93.57	5.15E-01	1.08E-02	96.46	186	151
NEA-GU-A Case 2														
LKF	0.00E+00	1.02E-01	3.00E-03	1.06E-01	4.70E-03	-4.22	1.18E-01	9.64E-03	-15.99	7.84E-02	6.03E-03	22.84	176	158
	2.00E-05	1.85E-01	3.72E-03	1.09E-01	6.40E-03	41.00	1.21E-01	9.58E-03	34.58	8.32E-02	7.54E-03	55.01	175	156
	2.00E-04	1.64E+00	3.99E-03	1.71E-01	7.24E-03	89.56	1.82E-01	1.69E-02	88.93	1.48E-01	8.83E-03	91.00	177	154
	2.00E-03	1.64E+01	3.42E-03	7.67E-01	6.56E-03	95.32	7.81E-01	1.11E-02	95.24	7.54E-01	7.37E-03	95.40	185	144
EKF	0.00E+00	1.02E-01	3.24E-03	1.06E-01	4.15E-03	-4.28	1.18E-01	9.21E-03	-15.78	6.37E-02	5.39E-03	37.46	176	157
	2.00E-05	1.86E-01	3.42E-03	1.11E-01	6.62E-03	40.42	1.22E-01	1.14E-02	34.56	6.77E-02	9.27E-03	63.55	175	157
	2.00E-04	1.64E+00	3.21E-03	1.71E-01	7.82E-03	89.58	1.79E-01	1.15E-02	89.10	1.29E-01	1.18E-02	92.17	178	154
	2.00E-03	1.64E+01	3.53E-03	7.68E-01	5.71E-03	95.32	7.87E-01	1.48E-02	95.20	7.17E-01	1.05E-02	95.63	185	144
UKF	0.00E+00	1.03E-01	2.59E-03	1.08E-01	4.33E-03	-4.85	1.22E-01	1.28E-02	-18.46	6.29E-02	5.61E-03	38.69	176	158
	2.00E-05	1.86E-01	2.93E-03	1.11E-01	6.47E-03	40.63	1.18E-01	7.81E-03	36.63	6.83E-02	1.03E-02	63.28	175	157
	2.00E-04	1.64E+00	3.26E-03	1.70E-01	6.81E-03	89.63	1.79E-01	1.19E-02	89.11	1.26E-01	1.02E-02	92.31	176	154
	2.00E-03	1.64E+01	3.73E-03	7.68E-01	6.65E-03	95.32	7.80E-01	9.28E-03	95.24	7.17E-01	9.82E-03	95.63	187	144

Table A-8. Performance results from NEA-GU-A. Number of runs = 25, Field of view = 0.8^{o}, Start time 11 s, End time 1037 s

Scenario: Northeast Asia to Guam with descending orbit (NEA-GU-D)

This scenario utilizes two sensor satellites in a 1350 km altitude 58 degree inclination descending node orbit. The target missile is an ICBM class missile launched from Northeast Asia targeting Guam. The missile flight time is 20 minutes. Figure A-49 is a diagram of the sensor satellites and target trajectories.

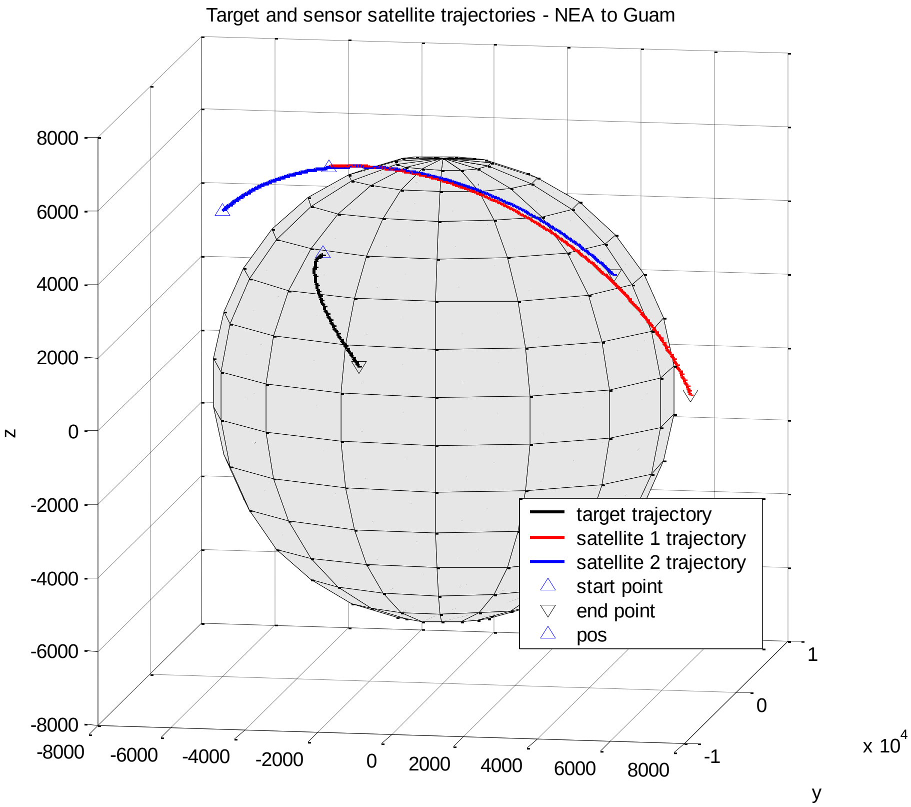


Figure A-49. Satellite and target for NEA-GU-D scenario

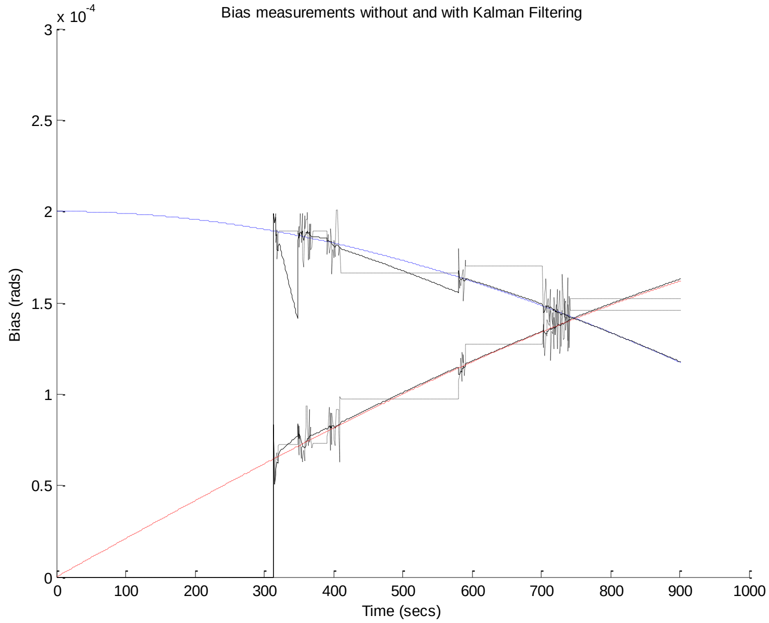


Figure A-50. Case 1: Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias

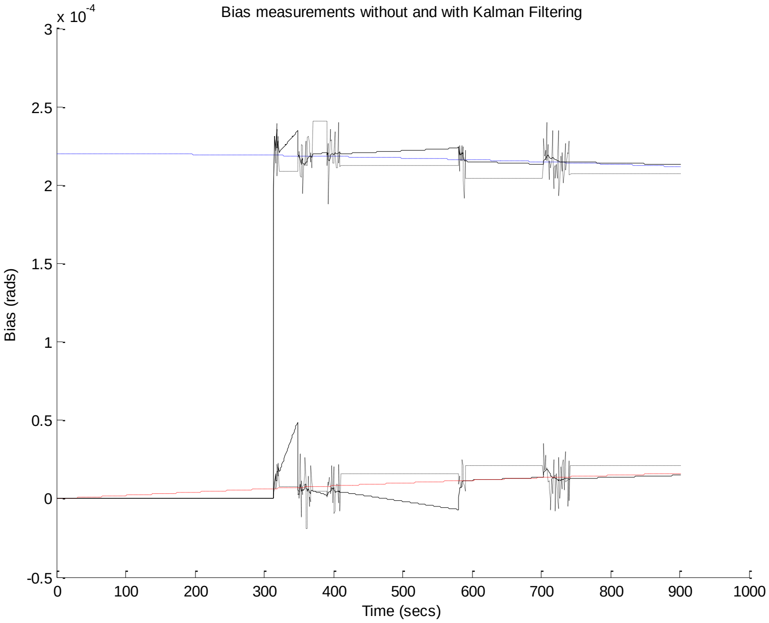


Figure A-51. Case 2: Sensor 1 bias measurements and estimates with and without bias EKF for constant bias with sinusoidal component.

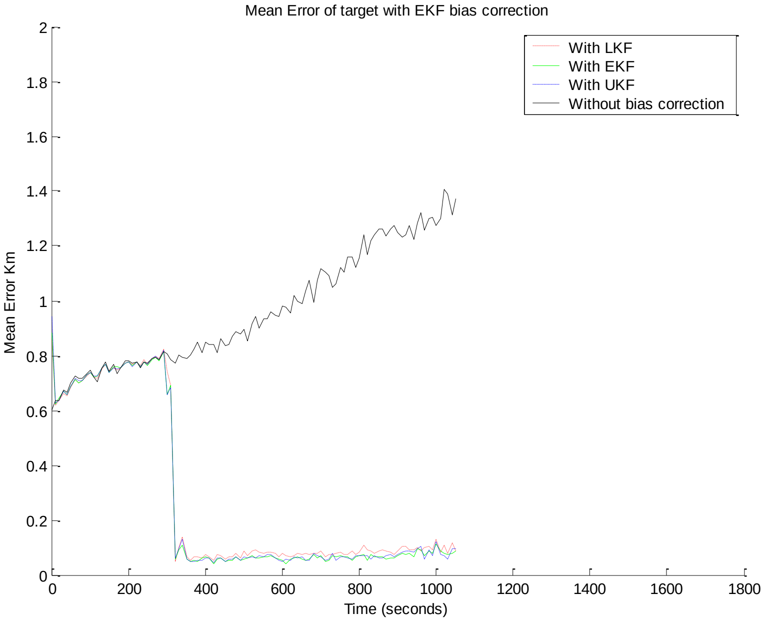


Figure A-52. Case 1: Mean error of target position for sinusoidal bias (β_0 = .2 *m*rads)

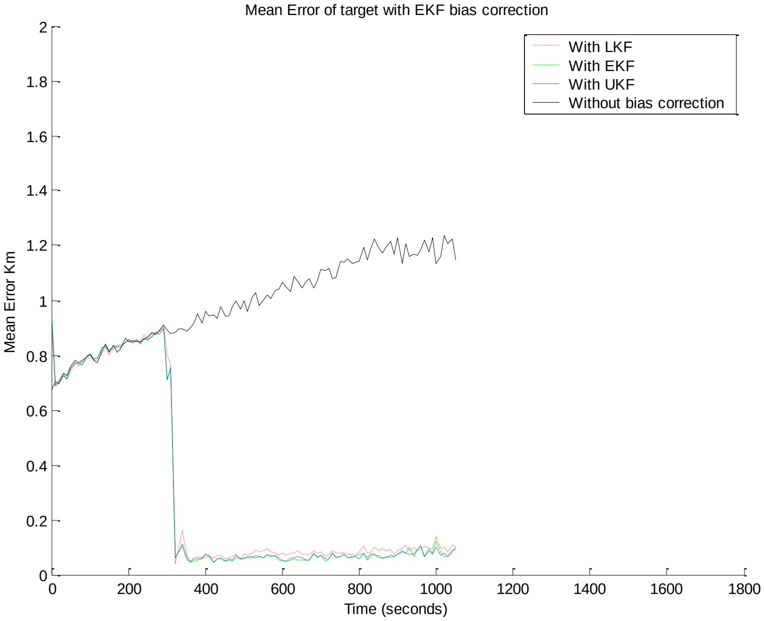


Figure A-53. Case 2: Bias measurements and estimates with and without bias EKF for constant bias with sinusoidal component.

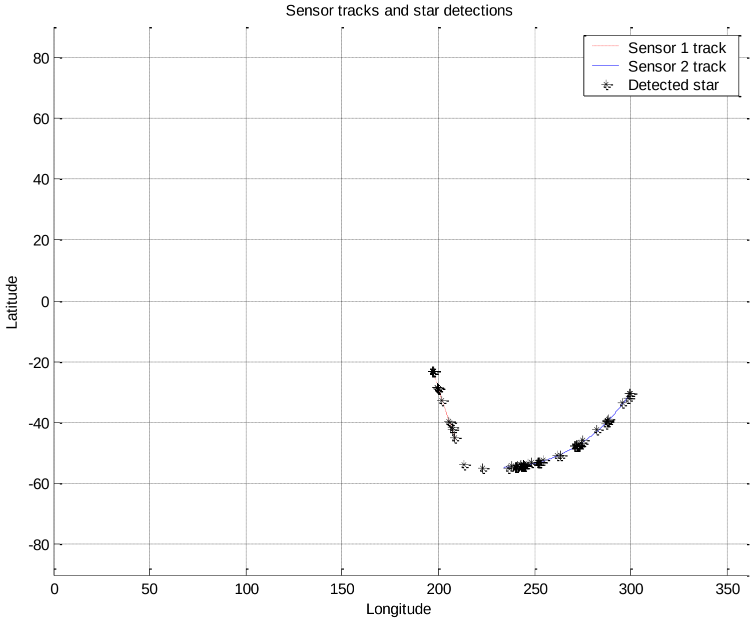


Figure A-54. Sensor trajectories with detected stars for NEA-GU-D scenario

NEA-GU-D Case 1														
Filter type		without bias correction		with EKF bias correction			with Sample and Hold correction			With EKF bias correction and post filtering			Numbe of stars	
	bias	Mean Err	STD	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Sensor 1	Sensor 2
LKF	0.00E+00	9.37E-02	1.54E-03	1.00E-01	3.64E-03	-6.91	1.10E-01	9.67E-03	-17.62	7.68E-02	4.77E-03	18.07	175	326
	2.00E-05	1.35E-01	2.14E-03	1.01E-01	3.35E-03	24.88	1.13E-01	1.36E-02	16.21	7.79E-02	3.54E-03	42.21	174	326
	2.00E-04	9.85E-01	2.31E-03	1.17E-01	4.71E-03	88.16	1.50E-01	1.07E-02	84.80	9.23E-02	6.71E-03	90.63	176	326
	2.00E-03	9.80E+00	1.92E-03	2.53E-01	2.91E-03	97.42	8.65E-01	2.80E-02	91.17	2.36E-01	3.39E-03	97.59	178	309
EKF	0.00E+00	9.40E-02	1.38E-03	1.03E-01	4.87E-03	-9.08	1.15E-01	1.38E-02	-22.66	6.57E-02	6.71E-03	30.09	174	326
	2.00E-05	1.35E-01	1.89E-03	1.03E-01	3.23E-03	23.93	1.13E-01	1.22E-02	16.40	6.47E-02	4.22E-03	52.05	174	326
	2.00E-04	9.85E-01	1.55E-03	1.17E-01	3.96E-03	88.12	1.52E-01	1.35E-02	84.56	7.96E-02	4.79E-03	91.92	176	326
	2.00E-03	9.80E+00	1.80E-03	2.53E-01	4.28E-03	97.41	8.68E-01	2.71E-02	91.14	2.33E-01	5.26E-03	97.62	177	310
UKF	0.00E+00	9.45E-02	2.19E-03	1.03E-01	4.36E-03	-8.68	1.11E-01	1.22E-02	-17.45	6.42E-02	4.82E-03	32.02	175	325
	2.00E-05	1.34E-01	1.70E-03	1.02E-01	4.65E-03	24.28	1.12E-01	9.88E-03	16.44	6.44E-02	4.81E-03	52.02	175	327
	2.00E-04	9.85E-01	1.78E-03	1.16E-01	4.37E-03	88.24	1.49E-01	1.44E-02	84.89	7.99E-02	5.35E-03	91.89	176	327
	2.00E-03	9.80E+00	1.72E-03	2.52E-01	2.71E-03	97.43	8.63E-01	2.74E-02	91.19	2.31E-01	3.18E-03	97.64	178	310
NEA-GU-D Case 2														
LKF	0.00E+00	9.41E-02	1.50E-03	1.01E-01	2.77E-03	-6.88	1.11E-01	1.35E-02	-18.42	7.67E-02	3.52E-03	18.52	174	326
	2.00E-05	1.41E-01	1.84E-03	1.02E-01	3.86E-03	27.17	1.12E-01	1.01E-02	20.60	7.92E-02	4.62E-03	43.67	175	326
	2.00E-04	1.04E+00	1.74E-03	1.19E-01	5.12E-03	88.53	1.28E-01	1.09E-02	87.64	9.63E-02	7.05E-03	90.73	177	327
	2.00E-03	1.03E+01	1.75E-03	2.56E-01	3.90E-03	97.52	2.88E-01	1.39E-02	97.21	2.43E-01	5.89E-03	97.65	185	352
EKF	0.00E+00	9.46E-02	2.15E-03	1.02E-01	4.59E-03	-8.06	1.12E-01	1.10E-02	-18.48	6.39E-02	6.00E-03	32.41	175	325
	2.00E-05	1.41E-01	2.24E-03	1.03E-01	3.43E-03	27.00	1.12E-01	9.53E-03	20.21	6.37E-02	3.85E-03	54.64	175	326
	2.00E-04	1.04E+00	2.09E-03	1.17E-01	2.76E-03	88.75	1.27E-01	9.49E-03	87.78	8.08E-02	3.65E-03	92.22	177	327
	2.00E-03	1.03E+01	1.77E-03	2.54E-01	3.33E-03	97.54	2.91E-01	1.68E-02	97.18	2.36E-01	3.61E-03	97.72	185	353
UKF	0.00E+00	9.45E-02	1.91E-03	1.02E-01	4.27E-03	-7.99	1.13E-01	9.20E-03	-19.98	6.46E-02	3.95E-03	31.68	175	326
	2.00E-05	1.40E-01	1.42E-03	1.04E-01	4.22E-03	25.96	1.13E-01	1.05E-02	19.21	6.52E-02	5.60E-03	53.46	175	325
	2.00E-04	1.04E+00	1.59E-03	1.17E-01	2.56E-03	88.71	1.28E-01	9.25E-03	87.68	8.04E-02	3.45E-03	92.26	177	327
	2.00E-03	1.03E+01	1.63E-03	2.56E-01	3.22E-03	97.52	2.89E-01	1.46E-02	97.21	2.38E-01	5.49E-03	97.70	185	352

Table A-9. Performance results from NEA-GU-D. Number of runs = 25, Field of view = 0.8°, Start time 0 s, End time 1060 s

Scenario: Iridium Satellite tracking with ascending orbit (NEA-IR-A)

This scenario utilizes two sensor satellites in a 1350 km altitude 58 degree inclination ascending node orbit. The target is an Iridium communications satellite in a low earth orbit.. The satellite is tracked for 30 minutes. Figure A-55 is a diagram of the sensor and target satellite trajectories.

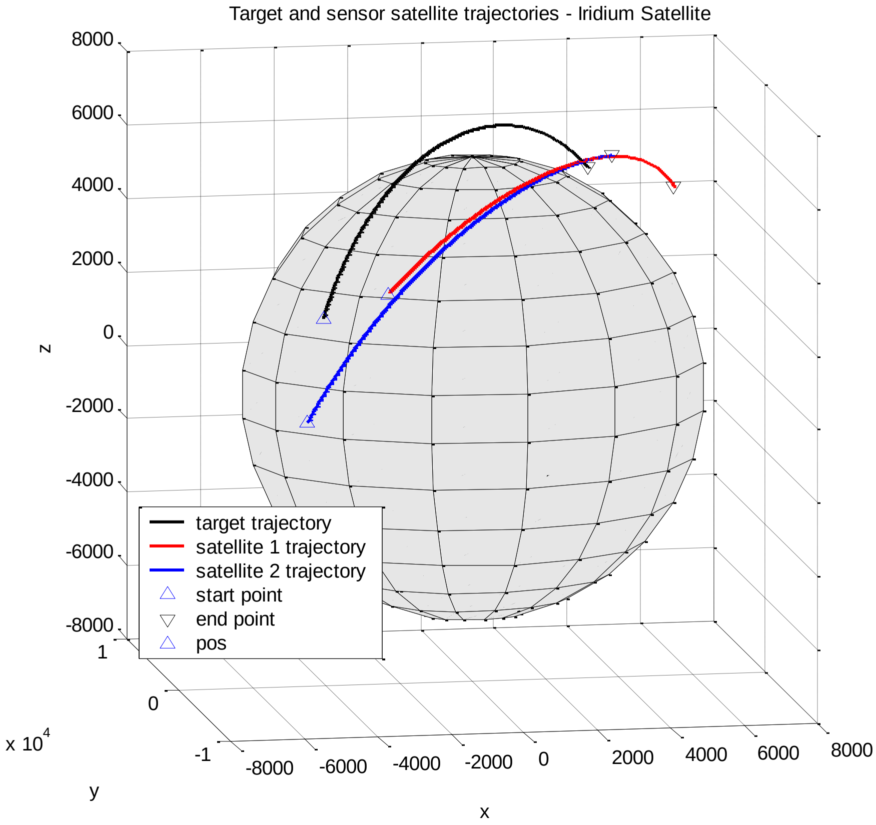


Figure A-55. Satellite and target trajectories for NEA-IR-A scenario

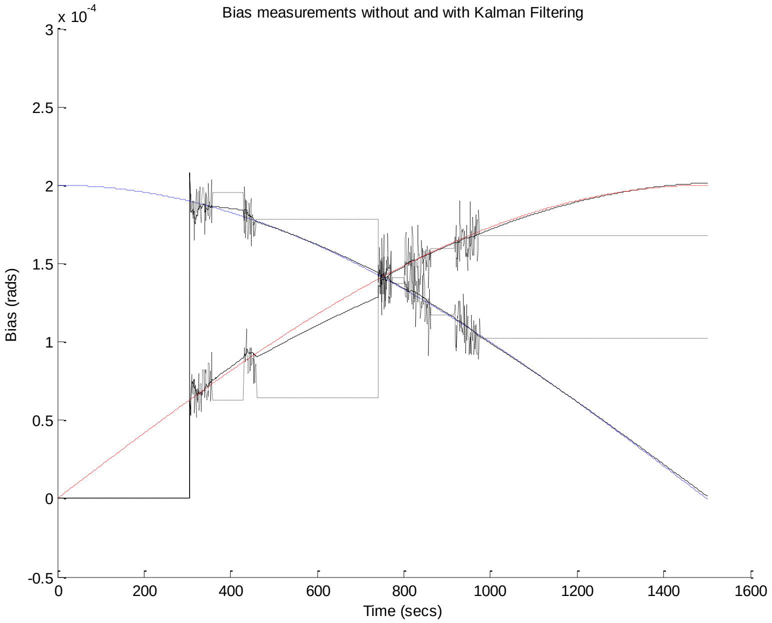


Figure A-56. Case 1: Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias

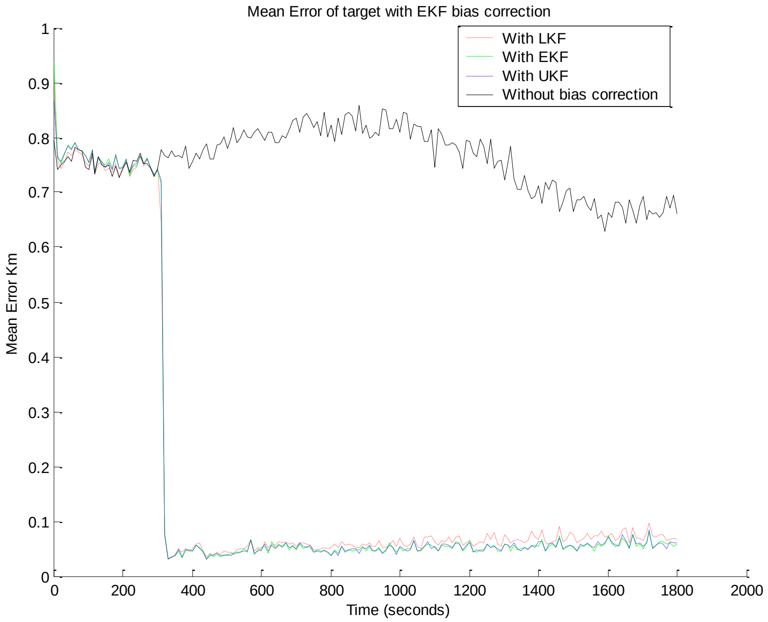


Figure A-57. Case 1: Mean error of target position for sinusoidal bias (β_0 = .2 *m*rads)

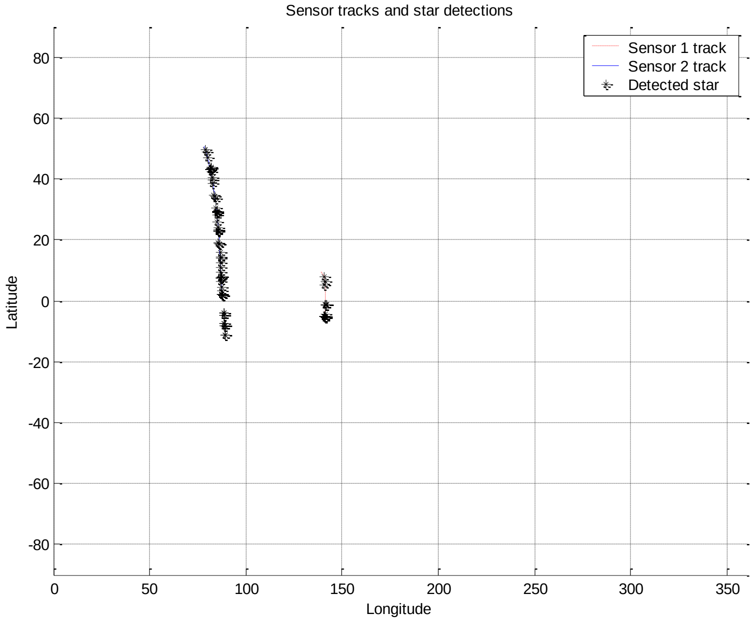


Figure A-58. Sensor trajectories with detected stars for NEA-IR-A scenario

Scenario: Iridium Satellite tracking with descending orbit (NEA-IR-D)

This scenario utilizes two sensor satellites in a 1350 km altitude 58 degree inclination descending node orbit. The target is an Iridium communications satellite in a low earth orbit.. The satellite is tracked for 30 minutes. Figure A-59 is a diagram of the sensor and target satellite trajectories.

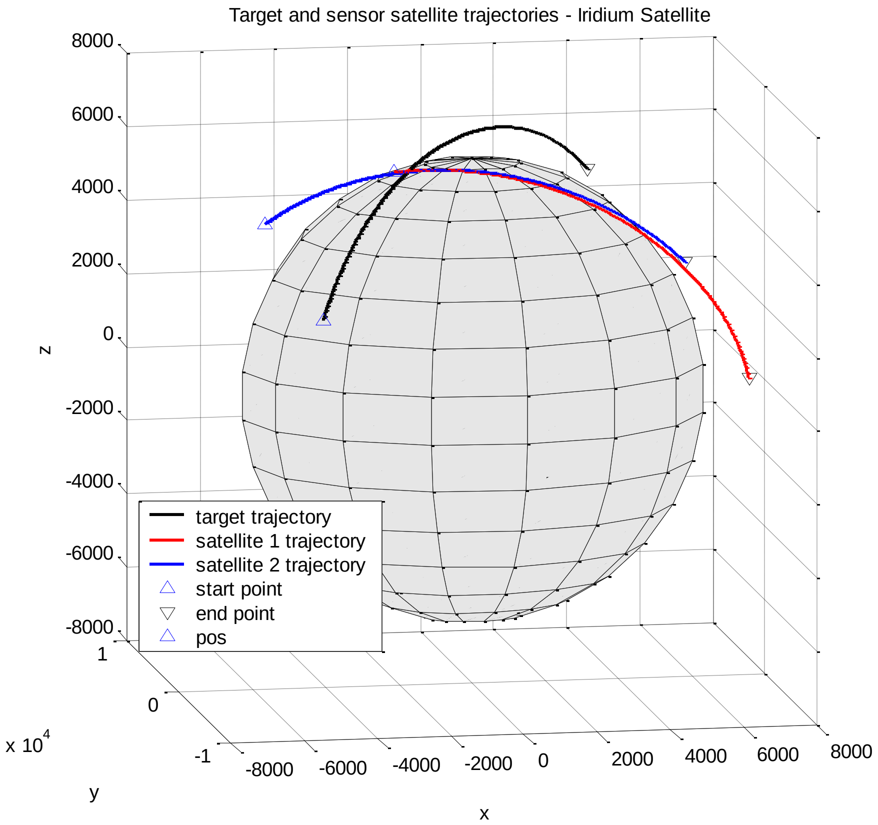


Figure A-59. Satellite and target trajectories for NEA-IR-D scenario

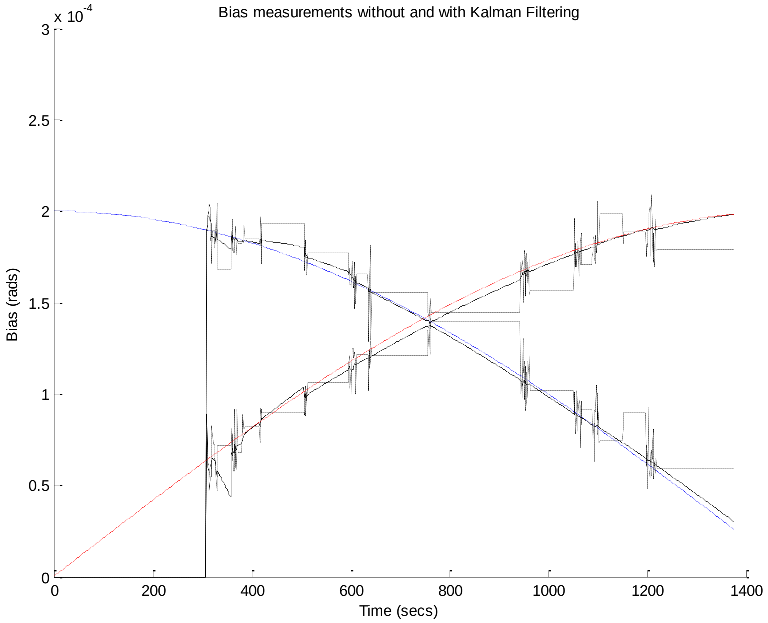


Figure A-60. Case 1: Sensor 1 bias measurements and estimates with and without bias EKF for pure sinusoidal bias

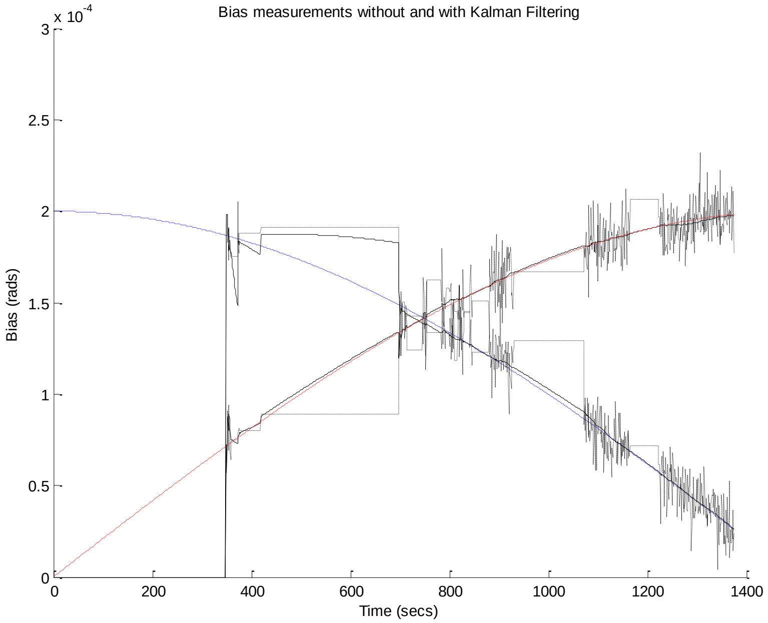


Figure A-61. Case 1: Sensor 2 bias measurements and estimates with and without bias EKF for pure sinusoidal bias

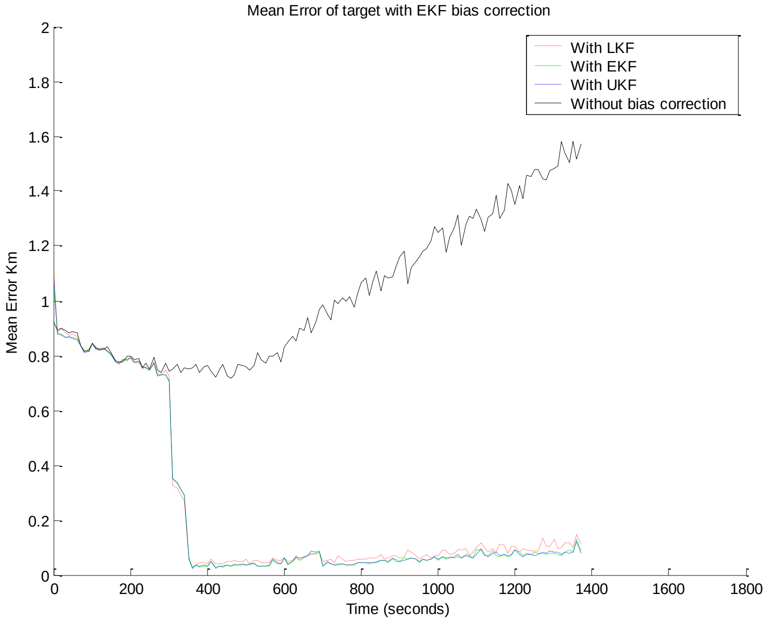


Figure A-62. Case 1: Mean error of target position for sinusoidal bias (β_0 = .2 *m*rads)

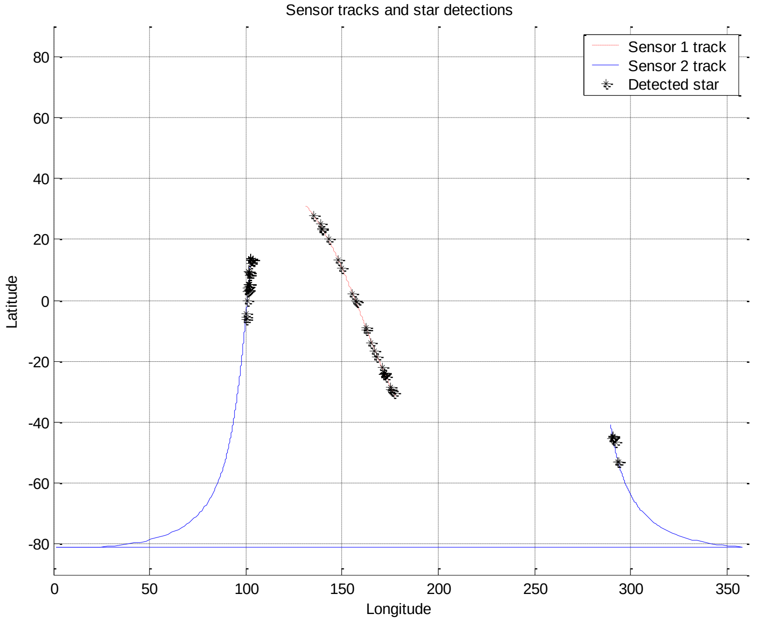


Figure A-63. Sensor trajectories with detected stars for NEA-IR-D scenario

NEA-IR-A Case 1														
Filter type		without bias correction		with EKF bias correction			with Sample and Hold correction			With EKF bias correction and post filtering			Numbe of stars	
	bias	Mean Err	STD	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Mean Err	STD	Pct Impr	Sensor 1	Sensor 2
LKF	0.00E+00	7.45E-02	7.96E-04	7.56E-02	1.15E-03	-1.53	8.46E-02	6.11E-03	-13.61	5.82E-02	1.87E-03	21.86	319	679
	2.00E-05	1.05E-01	9.43E-04	7.56E-02	1.55E-03	28.09	8.54E-02	8.29E-03	18.72	5.77E-02	1.06E-03	45.05	321	681
	2.00E-04	7.59E-01	9.03E-04	8.39E-02	1.41E-03	88.94	2.36E-01	1.64E-02	68.97	6.86E-02	1.83E-03	90.96	316	669
	2.00E-03	7.55E+00	7.89E-04	1.97E-01	3.73E-03	97.39	2.10E+00	2.12E-02	72.22	1.95E-01	4.25E-03	97.42	265	628
EKF	0.00E+00	7.45E-02	8.66E-04	7.57E-02	1.24E-03	-1.58	8.45E-02	7.16E-03	-13.43	4.87E-02	1.94E-03	34.63	320	680
	2.00E-05	1.05E-01	9.57E-04	7.55E-02	1.22E-03	27.98	8.85E-02	7.87E-03	15.62	4.89E-02	1.67E-03	53.39	320	679
	2.00E-04	7.59E-01	1.05E-03	8.36E-02	1.14E-03	88.99	2.33E-01	1.75E-02	69.37	5.94E-02	1.63E-03	92.18	316	667
	2.00E-03	7.55E+00	8.92E-04	1.99E-01	5.27E-03	97.37	2.09E+00	2.28E-02	72.31	1.86E-01	4.32E-03	97.54	266	628
UKF	0.00E+00	7.41E-02	8.56E-04	7.52E-02	1.26E-03	-1.47	8.46E-02	7.89E-03	-14.24	4.79E-02	1.68E-03	35.31	320	679
	2.00E-05	1.05E-01	8.30E-04	7.57E-02	9.91E-04	27.96	8.63E-02	9.08E-03	17.91	4.88E-02	1.67E-03	53.60	319	679
	2.00E-04	7.59E-01	8.13E-04	8.36E-02	1.40E-03	88.98	2.37E-01	1.58E-02	68.80	5.93E-02	1.92E-03	92.19	316	228
	2.00E-03	7.55E+00	8.76E-04	1.99E-01	3.69E-03	97.37	2.09E+00	1.96E-02	72.30	1.86E-01	3.28E-03	97.54	266	628
NEA-IR-D Case 1														
LKF	0.00E+00	9.98E-02	1.55E-03	1.03E-01	1.71E-03	-3.32	1.15E-01	1.05E-02	-15.11	7.02E-02	3.18E-03	29.71	181	774
	2.00E-05	1.45E-01	1.92E-03	1.04E-01	2.80E-03	28.51	1.19E-01	9.22E-03	17.94	7.10E-02	3.91E-03	51.09	181	774
	2.00E-04	1.08E+00	1.79E-03	1.18E-01	2.41E-03	89.08	1.53E-01	1.15E-02	85.81	8.67E-02	4.89E-03	91.97	177	767
	2.00E-03	1.08E+01	1.57E-03	2.77E-01	4.39E-03	97.42	8.14E-01	1.70E-02	92.43	2.57E-01	5.53E-03	97.61	185	702
EKF	0.00E+00	9.92E-02	1.66E-03	1.03E-01	2.94E-03	-3.33	1.15E-01	8.39E-03	-16.32	5.61E-02	4.71E-03	43.43	182	775
	2.00E-05	1.45E-01	1.71E-03	1.04E-01	2.44E-03	27.88	1.16E-01	8.26E-03	19.75	5.80E-02	3.92E-03	59.91	182	775
	2.00E-04	1.08E+00	1.55E-03	1.17E-01	1.86E-03	89.15	1.54E-01	1.15E-02	85.77	7.23E-02	2.86E-03	93.30	177	768
	2.00E-03	1.08E+01	1.73E-03	2.78E-01	3.20E-03	97.42	8.16E-01	2.05E-02	92.42	2.43E-01	5.44E-03	97.74	185	698
UKF	0.00E+00	9.95E-02	1.71E-03	1.03E-01	2.46E-03	-3.04	1.15E-01	7.15E-03	-15.26	5.68E-02	4.15E-03	42.91	181	775
	2.00E-05	1.44E-01	1.86E-03	1.03E-01	2.76E-03	28.56	1.13E-01	6.23E-03	21.77	5.75E-02	3.44E-03	60.16	181	771
	2.00E-04	1.08E+00	1.41E-03	1.18E-01	2.67E-03	89.11	1.58E-01	1.08E-02	85.42	7.32E-02	3.81E-03	93.22	177	768
	2.00E-03	1.08E+01	1.57E-03	2.78E-01	3.99E-03	97.41	8.12E-01	1.76E-02	92.45	2.45E-01	5.71E-03	97.72	185	698

Table A-10. Performance results from NEA-IR-A/D. Number of runs = 25, Field of view = 0.8^{o}, Start time s, End time

APPENDIX B: SELECTED MATLAB PROGRAM FILES

Base Tracking Program

```
%--- Initialize GSSM and variables
format short e;
global TS
global covsen
global covp

TS = 1;                          % time step
Nruns = 50             % input('Enter number of runs : ')
theatercase = 'NEA_SF_old'

covsen = (100.0e-6);         % sensor covarance
covbias = (2.0e-6);            % bias covariance
covp = .010;                     % process error
process_error = 1000*covp;

bias_senario_case = 1;   % Set bias scenario case: 1 = sinusoidal, 2 = constant w/sinusoid

biasvect = [0.00 20e-6 200e-6 2e-3]';      %Vector of bias values

w = 2*pi/6000;           % orbit frequency
fov = .5;              % sensor field of view in degrees
fovrad = fov*pi/180;     % sensor field of view in radians
app = cos(fovrad);       % cosine of apature size
Rearth = 6500;           % Earth radius in Km

QB = covbias^2;
RB = covsen^2 ;

savefile = fopen('C:/matlab_sv13/work/STproj_1/Output_Data/NEA_SF_old.txt', 'wt');

% write initial conditions to file
fprintf(savefile, 'Theater case:  %s\n', theatercase);
fprintf(savefile, 'Scenario case:  %s\n', scenario);
fprintf(savefile, 'Number of runs: %4.0f\n', Nruns);
fprintf(savefile, ‘Time Step: %4.0f\n', TS);
fprintf(savefile, 'Field of view: %6.2f\n', fov);
fprintf(savefile, 'Sensor covariance: %12.2e\n', covsen);
fprintf(savefile, 'Bias covariance: %12.2e\n', covsen);

%--- Generate inference data structure
model = gssm_tracking_8('init');
Arg.model = model;                              % embed GSSM
Arg.type = 'state';                             % estimation type
Arg.tag = 'State estimation for bearings-only tracking problem';  % info tag (not required)
InfDS = geninfds(Arg);                          % call generate function

% get data
% read star database
aura_norm;          % read star database normalized vectors (star  x, y, z)
NNtableindex;     % read nearest neighbor table Nearneigh
Nstars =  size(star.vector, 2);          % size of star database

% read target and satelite data
[target, Sat, Sensor] = get_NEA_data(theatercase);

target.true_state =       [target.posit(1,:);
                        target.vel(1,:);
                        target.posit(2,:);
                        target.vel(2,:);
                        target.posit(3,:);
                        target.vel(3,:)];

T = size(target.posit,2);          % Time span of track
Sensor(1).trktm = Sensor(1).trkend - Sensor(1).trkst + 1;        % Time span of observations
Sensor(2).trktm = Sensor(2).trkend - Sensor(2).trkst + 1;

kstart = max(Sensor(1).trkst, Sensor(2).trkst);         % observation start and end time
kend = min(Sensor(1).trkend, Sensor(2).trkend);

Tplot = kend-300;              % plot length

fprintf(savefile, 'Start time: %4.0f, End time: %4.0f\n', kstart, kend);      % write to file

% ----------------------------------

% Convert sensor LOS to radians

Sensor(1).los_rad = (pi/180.0).*Sensor(1).los_deg;
Sensor(2).los_rad = (pi/180.0).*Sensor(2).los_deg;

star.RAdec = vectobrg2(star.vector);            % Convert star ECI vector to right ascension and declination

% ------------ calculate bias waveform ---------------

if  bias_senario_case == 1
    scenario = 'sinusoidal'
      w0 = 0.001;
      for t = 1:T
         bsaz(:,t) =   biasvect*cos(w*t);                     % azimuth bias
         bsel(:,t) =   biasvect*cos(w*t + 3*pi/2);             % elevation bias
      end              % t
end        %bias_senario_case = 1

if  bias_senario_case == 2
    scenario = 'const w/sinusoid'
     w0 = 0.00;
       for t = 1:T
          bsaz(:,t) =   biasvect + .1*biasvect*cos(w*t);
          bsel(:,t) =   .1*biasvect*cos(w*t + 3*pi/2);
    end % t
end        % bias_senario_case = 2

Sensor(1).bias_true = zeros(4,T);
Sensor(2).bias_true = zeros(4,T);

% ---------------------  vary filter type   ---------------------

litype = {'r:','g-', 'b-.', 'k--'};     % line type for different graphs
lftype = {'kf','ekf', 'ukf'};

for kk =1:1                        % kk = input('Enter filter case (1 = kf, 2 = EKF, 3 = UKF : ')

  lk = litype{kk};                 % line type by filter type
  ftype = lftype{kk}
  fprintf(savefile, '\r Filter type:%s\n', ftype);

[pNoise, oNoise, InfDS] = gensysnoiseds(InfDS, ftype);     % call system noise sources generation function

% --------------------  % vary bias value  ----------------------

for b = 3:3           % b = input('Enter bias value 1 = 0.0, 2 = 20e-6, 3 = 200e-6, 4 = 2e-3 : ')
 bias_amount = biasvect(b)

    Sensor(1).bias_true = [bsaz(b,:); bsel(b,:)];
    Sensor(2).bias_true = [bsaz(b,:); bsel(b,:)];

    li = litype{b};      % line type by bias value

% ------------------------------------------------------------------------
% -------------------  Cycle through number of runs  ----------------------
% ------------------------------------------------------------------------
for nr = 1:Nruns

nstars_1=0;               % initialize star counter for sensor 1
nstars_2=0;               % initialize star counter for sensor 2
Tfirst1 = 1;              % initialize star interval counter flags
Tfirst2 = 1;
newtrack = 1;             % set newtrack flag

target.state_est = zeros(InfDS.statedim, T);      % initialize state estimate arrays
target.sen1_state_est = target.state_est;
target.sen2_state_est = target.state_est;
Xh_ = target.state_est;
Px = eye(InfDS.statedim);          % initial state covariances
Px_1 = Px;
Px_2 = Px;

V = feval(model.pNoise.sample, model.pNoise, T);                                   % generate process noise
Sensor(1).noise = feval(model.oNoise.sample, model.oNoise, Sensor(1).trktm);     % generate observation
noise sensor 1
Sensor(2).noise = feval(model.oNoise.sample, model.oNoise, Sensor(2).trktm);     % generate observation
noise sensor 2
```

```
% -------------------- Calculate noisy measurements ----------------------------------

% add bias
y1biased(:,:) = addangle(Sensor(1).los_rad(:,:), Sensor(1).bias_true(:,Sensor(1).trkst:Sensor(1).trkend));
y2biased(:,:) = addangle(Sensor(2).los_rad(:,:), Sensor(2).bias_true(:,Sensor(2).trkst:Sensor(2).trkend));

% add observation noise to sensor measurements
Sensor(1).obs_bodyang_noisy(:,:) = addangle(y1biased(:,:), Sensor(1).noise(:,:));
Sensor(2).obs_bodyang_noisy(:,:) = addangle(y2biased(:,:), Sensor(2).noise(:,:));

% Convert azimuth and elevation measurements to boresight vectors
Sensor(1).obs_bodyvec_noisy(:,:) = brngtovec(Sensor(1).obs_bodyang_noisy(:,:));
Sensor(2).obs_bodyvec_noisy(:,:) = brngtovec(Sensor(2).obs_bodyang_noisy(:,:));
test = Sensor(1).obs_bodyvec_noisy(:,:);

% convert boresight vectors to ECI coordinate frame using satellite attitude quaterons
Sensor(1).obs_ECIvec_noisy(:,:) = quatxfrm(Sat(1).att(:,Sensor(1).trkst:Sensor(1).trkend), …
Sensor(1).obs_bodyvec_noisy(:,:));
Sensor(2).obs_ECIvec_noisy(:,:) = quatxfrm(Sat(2).att(:,Sensor(2).trkst:Sensor(2).trkend),  …
Sensor(2).obs_bodyvec_noisy(:,:));

% convert ECI vector to ECI azimuth and elevation in degrees
Sensor(1).obs_ECIae_noisy(:,:) = vectobrg2(Sensor(1).obs_ECIvec_noisy(:,:));
Sensor(2).obs_ECIae_noisy(:,:) = vectobrg2(Sensor(2).obs_ECIvec_noisy(:,:));

%------------------- initial bias estimate ------------------------
Sensor(1).bias_sh(:,1) = [0 0]';
Sensor(2).bias_sh(:,1) = [0 0]';

% initialize bias filter state vectors
xBh1a = zeros(3,T);     % sensor 1 azimuth
xBh1e = zeros(3,T);     % sensor 1 elevation
xBh2a = zeros(3,T);     % sensor 2 azimuth
xBh2e = zeros(3,T);     % sensor 2 elevation

xBh1a(:,1) = [0 0 w0]';
xBh1e(:,1) = [0 0 w0]';
xBh2a(:,1) = [0 0 w0]';
xBh2e(:,1) = [0 0 w0]';

PB1a = eye(3);      % initial bias state vector covariance matricies
PB1e = eye(3);
PB2a = eye(3);
PB2e = eye(3);

Sensor(1).bias_est = zeros(2,T);   % initial bias estimates
Sensor(2).bias_est = zeros(2,T);

Sensor(1).intrack = 0;                    % set sensor track flag
Sensor(2).intrack = 0;

Tcount1 = 0;        % initialize time between star observation counters
```

```
Tcount2 = 0;

% ----------------------------------------------------------------
% -------------------Step through time ------------------------
% ----------------------------------------------------------------
for t = 1:TS:T
time = t ;
star_case = [0 0]';     % reset star detection flag

% ---------------------------------------------------------------------
% ---------------------- Sensor 1 tracking ---------------------------
% ---------------------------------------------------------------------
if (t >= Sensor(1).trkst) && (t <= Sensor(1).trkend)   % determine if sensor is tracking target
   Sensor(1).intrack = 1;              % sensor 1 is tracking
   k1 = t - Sensor(1).trkst + 1;       % update sensor tracking counter

% ---------------------- Star Check for sensor 1 -------------------------------
if t > 299

% ----------- check if sensor LOS is blocked by earth ----------------

Sat(1).alt = sqrt(Sat(1).posit(1,t)^2 + Sat(1).posit(2,t)^2 + Sat(1).posit(3,t)^2); % sat altitude
Sat(1).earthvect = -Sat(1).posit(:,t)/Sat(1).alt;       % unit vector from sat to earth center
thetaLOS1 = acos(dot(Sensor(1).obs_ECIvec_noisy(:,k1), Sat(1).earthvect));       % angle between satellite
fov and earth center
thetaearth1 = asin(Rearth/Sat(1).alt);                  % angle between satellite and earth rim
 R1 = Sat(1).alt*cos(thetaearth1);

if thetaearth1 > thetaLOS1       % if LOS is within earth radius
   star_case(1) = 0;                 % then cannot see star
else
%-----------------------------------------------------------------------
% if not blocked find nearest star
%-----------------------------------------------------------------------
if Tcount1 >= 60/TS          % if a star hasn't been seen in 60 seconds
   Tfirst1 = 1;                     % set nearest star algorithm flag
end

if Tfirst1 == 1                    % if first time through
   for i1 = 1:Nstars            % calculate distance from observation vector to each star in catalog
     dist1(i1) = dot(star.vector(:,i1),Sensor(1).obs_ECIvec_noisy(:,t));   % distance = dot product of vectors
   end     % end i1

   % sort distances. Index number = star number in star catalog
   [sort_dist1, S_index_1] = sort(dist1);

   % Select nearest star (greatest value of distance is closest)
   nearest_1 = S_index_1(Nstars);

   % clear unneeded arrays
   clear sort_dist1;
   clear dist1;
   clear S_index_1;
   Tfirst1 = 0;    % reset flag
   Tcount1 = 0;    % and counter
end     % end if Tfirst = 1

% ---------------------------------------------
% compare star vector to observation vector for 11 nearest stars
% -----------------------------------------------
j1=0;  % reset observed star counter

for i = 1:11    % for each star in nearest neighbor table assign NNs1 as star number

   NNs1 = Nearneigh(i,nearest_1);

% calculate distance from observation vector to each star
   dist1 = dot(star.vector(:,NNs1),Sensor(1).obs_ECIvec_noisy(:,k1));

% build tables of stars and distances to vectors
   NNtable_1(:,i) = [NNs1, dist1]';

% ---  Check if star is in sensor 1 view  --------
   if dist1 > app
      star_case(1) = 1;   % a star is detected
      nstars_1 = nstars_1+1;            % running number of stars detected
      j1=j1+1;            % number of stars detected in this time step
      t1(nstars_1) = t;          % time of detection
      starplot1(:,nstars_1) = [star.RAdec(1,NNs1) star.RAdec(2,NNs1) NNs1 t]'; % detected star location ID
number and time

% ------- Find bias from expected and actual star observations ---------
% find true star position in sensor frame (azimuth and elevation)
star_vec_bf = invquatxfrm(Sat(1).att(:,t), star.vector(:,NNs1));  % Rotate star vector to sensor frame
star_azel_bf = vectobrg(star_vec_bf)*pi/180;  % express in azimuth and elevation in radians

% calculate actual observed star azimuth and elevation
angledelta_1(:,nstars_1) = subangle(star_azel_bf, Sensor(1).obs_bodyang_noisy(:,k1));
% Calculate expected star observation
angledelta_exp_1(:,nstars_1) = subangle(star_azel_bf, Sensor(1).los_rad(:,k1));

% calculate observed bias
Sensor(1).bias_obs(:,nstars_1) = subangle(angledelta_exp_1(:,nstars_1), angledelta_1(:,nstars_1));   %
azimuth
% Sensor(1).bias_obs(2,nstars_1) = subangle(angledelta_exp_1(2,nstars_1), angledelta_1(2,nstars_1));   %
elevation

   end     % end dist1

end     % end i

%  ----------------------------------

% reset nearest stars by sorting by distance

[newNNtable_1, Q] = sort(NNtable_1(2,:));
```

```
nearest_1 = NNtable_1(1,Q(11));

end     % end earth interference check

end     % end if t > 299
else
   Sensor(1).intrack = 0;      % sensor 1 is not in track
end     % end intrack

% -----------------------------------------------------------
% ----------------- Bias Filter -----------------------
% ------------------------------------------------------------
if t > 1
   S1 = 0;

if star_case(1) == 0    % no star detected
   %  for simple bias estimate -  use previous bias observation
   Sensor(1).bias_sh(:,t) = Sensor(1).bias_sh(:,t-TS);

   xBh1a(:,t) = project2(1, xBh1a(:,t-TS), 100);    % project azimuth estimate forward
   xBh1e(:,t) = project2(1, xBh1e(:,t-TS), 100);    % project elevation estimate forward

   % for EKF bias - use predicted bias estimate
   Sensor(1).bias_est(1,t) =  xBh1a(1,t);
   Sensor(1).bias_est(2,t) =  xBh1e(1,t);

   Tcount1 = Tcount1 + 1;      % increment observation interval counter

elseif star_case(1) == 1    % star is detected
      Tcount1 = 0;        % reset observation interval counter
     for j = 1:j1
      S1 = S1 + Sensor(1).bias_obs(:,nstars_1-j+1);     % if more than one star is identified in view, average
the bias observations
     end     %end j
   Sensor(1).bias_sh(:,t) = S1/j1;   % average bias measurement
    ZB1 = S1/j1;

       dn = nstars_1-j1;
     if dn == 0;
        dt1 = TS;
     else
      dt1 = t1(nstars_1) - t1(dn);
   end

  % run bias EKF
   [xBh1a(:,t), PB1a_] = Bias_KF(dt1, xBh1a(:,t-dt1), PB1a, RB, ZB1(1), QB);

   [xBh1e(:,t), PB1e_] = Bias_KF(dt1, xBh1e(:,t-dt1), PB1e, RB, ZB1(2), QB);
   Sensor(1).bias_est(1,t) = xBh1a(1,t);    % EKF azimuth bias estimate
   Sensor(1).bias_est(2,t) = xBh1e(1,t);    % EKF elevation bias estimate
   PB1a = PB1a_;
   PB1e = PB1e_;
```

```
else
    errorstatement = 'star_case error'
end     % end if star_case == 1

end     % end if t>1

% ---------------------------------------------------------------------
% -------------------------- Sensor 2 tracking  -------------------------
% ---------------------------------------------------------------------
if (t >= Sensor(2).trkst) && (t <= Sensor(2).trkend)
    Sensor(2).intrack = 1;
    k2 = t - Sensor(2).trkst + 1;

% ---------------------- Star Check for sensor 2 -------------------------------
if t > 299

% check if sensor LOS is blocked by earth

Sat(2).alt = sqrt(Sat(2).posit(1,t)^2 + Sat(2).posit(2,t)^2 + Sat(2).posit(3,t)^2);  % sat altitude
Sat(2).earthvect = -Sat(2).posit(:,t)/Sat(2).alt;       % unit vector from sat to earth center
thetaLOS2 = acos(dot(Sensor(2).obs_ECIvec_noisy(:,k2), Sat(2).earthvect));       % angle between satellite
fov and earth center
thetaearth2 = asin(Rearth/Sat(2).alt);                   % angle between satellite and earth rim

if thetaearth2 > thetaLOS2        % if LOS is within earth radius
    star_case(2) = 0;                  % then cannot see star

else

%------------------------------------------------------------------------
% find nearest star for first observation
%------------------------------------------------------------------------
if Tcount2 >= 60/TS
    Tfirst2 = 1;
%     timecount60_2 = t;
end

if Tfirst2 == 1
for i2 = 1:Nstars         % calculate distance from observation vector to each star in catalog
       % distance = dot product of vectors
    dist2(i2) = dot(star.vector(:,i2),Sensor(2).obs_ECIvec_noisy(:,k2));
end % end i2

% sort distances. Index number = star number in star catalog
[sort_dist2, S_index_2] = sort(dist2);

% Select nearest star (greatest value of distance is closest)
nearest_2=  S_index_2(Nstars);

% clear unneeded arrays
clear sort_dist2;
clear dist2;
clear S_index_2;
```

```
Tfirst2 = 0;
Tcount2 = 0;
end     % end if Tfirst2 = 1

% ---------------------------------------------
% compare star vector to observation vector for 11 nearest stars
% -----------------------------------------------

j2=0;  % reset observed star counter

for i = 1:11    % for each star in nearest neighbor table assign NNs1,2 as star number

   NNs2 = Nearneigh(i,nearest_2);

% calculate distance from observation vector to each star
   dist2 = dot(star.vector(:,NNs2),Sensor(2).obs_ECIvec_noisy(:,k2));

% build tables of stars and distances to vectors
   NNtable_2(:,i) = [NNs2, dist2]';

% ---  Check if stars are in sensor 2 view  --------
   if dist2 > app
      star_case(2) = 1;      % a star is detected
      nstars_2 = nstars_2+1;              % running number of stars detected
      j2 = j2+1;           % number of stars detected in this time step
      t2(nstars_2) = t;
     starplot2(:,nstars_2) = [star.RAdec(1,NNs2) star.RAdec(2,NNs2) NNs2 t]';  % detected star location ID
number and time

 % ------- Find bias from expected and actual star observations ---------

star_vec_bf = invquatxfrm(Sat(2).att(:,t), star.vector(:,NNs2));  % Rotate star vector to sensor frame
star_azel_bf = vectobrg(star_vec_bf)*pi/180; % express in azimuth and elevation in radians

% calculate actual observed star azimuth and elevation
angledelta_2(:,nstars_2) = subangle(star_azel_bf, Sensor(2).obs_bodyang_noisy(:,k2));

% Calculate expected star observation
angledelta_exp_2(:,nstars_2) = subangle(star_azel_bf, Sensor(2).los_rad(:,k2));

% calculate observed bias
Sensor(2).bias_obs(:,nstars_2) = subangle(angledelta_exp_2(:,nstars_2), angledelta_2(:,nstars_2)); %
azimuth
% bias_obs_2(2,nstars_2) = subangle(angledelta_exp_2(2,nstars_2), angledelta_2(2,nstars_2)); % elevation

   end   % end dist2

end % end i

% ----------------------------------------------------------

% reset nearest stars by sorting by distance
```

```
[newNNtable_2, Q] = sort(NNtable_2(2,:));
nearest_2 = NNtable_2(1,Q(11));

end     % end earth interference check
end     % end if t > 299
else
   Sensor(2).intrack = 0;
end     % end intrack
% ------------------------------------------------------------
% ----------------- Bias Filter -----------------------
% -------------------------------------------------------------

if t > 1
   S2 = 0;

if star_case(2) == 0  % no star detected

   %  for simple bias estimate -  use previous bias observation
   Sensor(2).bias_sh(:,t) = Sensor(2).bias_sh(:,t-TS);

   xBh2a(:,t) = project2(1, xBh2a(:,t-TS), 100);    % project azimuth estimate forward
   xBh2e(:,t) = project2(1, xBh2e(:,t-TS), 100);    % project elevation estimate forward

   % for EKF bias - use predicted bias estimate
   Sensor(2).bias_est(1,t) =  xBh2a(1,t);
   Sensor(2).bias_est(2,t) =  xBh2e(1,t);
   Tcount2 = Tcount2 + 1;

elseif star_case(2) == 1
   Tcount2 = 0;
    for j = 1:j2
      S2 = S2 + Sensor(2).bias_obs(1:2,nstars_2-j+1);  % if more than one star is identified in view, average
the bias observations
    end     %end j
    Sensor(2).bias_sh(:,t) = S2/j2;
    ZB2 = S2/j2;

      dm = nstars_2-j2;
    if dm == 0;
       dt2 = TS;
    else
      dt2 = t2(nstars_2) - t2(dm);
   end

  % run bias EKF
  [xBh2a(:,t), PB2a_] = Bias_KF(dt2, xBh2a(:,t-dt2), PB2a, RB, ZB2(1), QB);
  [xBh2e(:,t), PB2e_] = Bias_KF(dt2, xBh2e(:,t-dt2), PB2e, RB, ZB2(2), QB);
  Sensor(2).bias_est(1,t) = xBh2a(1,t);
  Sensor(2).bias_est(2,t) = xBh2e(1,t);
  PB2a = PB2a_;
  PB2e = PB2e_;
```

```
else
   errorstatement = 'star_case error'
end     % end if star_case == 1

end     % end if t>1

% ---------------------------------------------
% correct observation vectors with bias estimate
% --------------------------------------------

if Sensor(1).intrack == 1
% EKF bias
   Sensor(1).obs_bodyang_correct_EKF(:,k1) = subangle(Sensor(1).obs_bodyang_noisy(:,k1),
Sensor(1).bias_est(:,t));
   Sensor(1).obs_bodyvec_correct_EKF(:,k1) = brngtovec(Sensor(1).obs_bodyang_correct_EKF(:,k1));
% Put observation in Sensor vector form
   Sensor(1).obs_ECIvec_correct_EKF(:,k1) = quatxfrm(Sat(1).att(:,t),
Sensor(1).obs_bodyvec_correct_EKF(:,k1)); % Put observation in ECF vector form
   Sensor(1).obs_ECIang_correct_EKF(:,k1) = vectobrg2(Sensor(1).obs_ECIvec_correct_EKF(:,k1));    %
ECF bearing in radians

% sample and hold bias

   Sensor(1).obs_bodyang_correct_sh(:,k1) = subangle(Sensor(1).obs_bodyang_noisy(:,k1),
Sensor(1).bias_sh(:,t));
   Sensor(1).obs_bodyvec_correct_sh(:,k1) = brngtovec(Sensor(1).obs_bodyang_correct_sh(:,k1));          %
translate to vector form
   Sensor(1).obs_ECIvec_correct_sh(:,k1) = quatxfrm(Sat(1).att(:,t),
Sensor(1).obs_bodyvec_correct_sh(:,k1));   % rotate to ECF vector

end

if Sensor(2).intrack == 1
% EKF Bias
   Sensor(2).obs_bodyang_correct_EKF(:,k2) = subangle(Sensor(2).obs_bodyang_noisy(:,k2),
Sensor(2).bias_est(:,t));
   Sensor(2).obs_bodyvec_correct_EKF(:,k2) = brngtovec(Sensor(2).obs_bodyang_correct_EKF(:,k2));
   Sensor(2).obs_ECIvec_correct_EKF(:,k2) = quatxfrm(Sat(2).att(:,t),
Sensor(2).obs_bodyvec_correct_EKF(:,k2));
   Sensor(2).obs_ECIang_correct_EKF(:,k2) = vectobrg2(Sensor(2).obs_ECIvec_correct_EKF(:,k2));

% sample and hold bias

   Sensor(2).obs_bodyang_correct_sh(:,k2) = subangle(Sensor(2).obs_bodyang_noisy(:,k2),
Sensor(2).bias_sh(:,t));
   Sensor(2).obs_bodyvec_correct_sh(:,k2) = brngtovec(Sensor(2).obs_bodyang_correct_sh(:,k2));
   Sensor(2).obs_ECIvec_correct_sh(:,k2) = quatxfrm(Sat(2).att(:,t),
Sensor(2).obs_bodyvec_correct_sh(:,k2));

end
% --------------------------------------------------

bothintrack = Sensor(1).intrack + Sensor(2).intrack;
```

```
% ------------------------------------------------------------------
% --------------------- Filter Setup  ------------------------------
% ------------------------------------------------------------------

   A = model.A;
   G = model.G;

if bothintrack == 2


% calculate intercept point of the two vectors
target.intercept_nocorr(:,t) = intercept(Sensor(1).obs_ECIvec_noisy(:,k1),
Sensor(2).obs_ECIvec_noisy(:,k2), Sat(1).posit(:,t), Sat(2).posit(:,t));   % no bias correction
target.intercept_EKF(:,t) = intercept(Sensor(1).obs_ECIvec_correct_EKF(:,k1),
Sensor(2).obs_ECIvec_correct_EKF(:,k2), Sat(1).posit(:,t), Sat(2).posit(:,t));  % with EKF bias correction
target.intercept_sh(:,t) = intercept(Sensor(1).obs_ECIvec_correct_sh(:,k1),
Sensor(2).obs_ECIvec_correct_sh(:,k2), Sat(1).posit(:,t), Sat(2).posit(:,t));  % with simple bias correction

% random initial estimate
if newtrack == 1
   newtrack = 0;
% range to target from sensor 2
a2 = sqrt((target.intercept_EKF(1,1)-Sat(2).posit(1,1)).^2 + (target.intercept_EKF(2,1)-Sat(2).posit(2,1)).^2
+ (target.intercept_EKF(3,1)-Sat(2).posit(3,1)).^2);

target.state_est(1,t) = target.intercept_EKF(1,t) + 10*covsen*a2*randn(1);
target.state_est(3,t) = target.intercept_EKF(2,t) + 10*covsen*a2*randn(1);
target.state_est(5,t) = target.intercept_EKF(3,t) + 10*covsen*a2*randn(1);

target.sen1_state_est(:,t) = target.state_est(:,t);
target.sen2_state_est(:,t) = target.state_est(:,t);

% Initial covariance matrix for plotting
% P(1,1) = Px(1,1);
% P(2,1) = Px(3,3);
% P(3,1) = Px(5,5);

else

switch ftype
%-------------------  Linearized Kalman Filter ----------------------------------

case 'kf'       % kk = 1

[target.state_est(:,t), Px] = LKFts(model, pNoise, target.state_est(:,t-TS), Px, covsen,
Sensor(1).obs_bodyang_correct_EKF(:,k1), Sensor(2).obs_bodyang_correct_EKF(:,k2), Sat(1).posit(:,t),
Sat(2).posit(:,t), target.intercept_EKF(:,t));


%------------------- Extended Kalman Filter -----------------------------------
   case 'ekf'  % kk = 2
% find EKF estimate for each satellite sensor given corrected LOS
% observation and Satellite positions, using fused estimate from earlier time step
[target.sen1_state_est(:,t), Px_1] = ekf(target.state_est(:,t-TS), Px, pNoise, oNoise,
Sensor(1).obs_ECIang_correct_EKF(:,k1), [], Sat(1).posit(:,t), InfDS);
[target.sen2_state_est(:,t), Px_2] = ekf(target.state_est(:,t-TS), Px, pNoise, oNoise,
Sensor(2).obs_ECIang_correct_EKF(:,k2), [], Sat(2).posit(:,t), InfDS);

%------------------- Unscented Kalman Filter -----------------------------------
   case 'ukf'  % kk = 3

[pNoise, oNoise, InfDS] = gensysnoiseds(InfDS, ftype);     % call system noise sources generation function

      alpha = 1;         % scale factor (UKF parameter)
      beta  = 2;         % optimal setting for Gaussian priors (UKF parameter)
      kappa = 2;         % optimal for state dimension=2 (UKF parameter)

InfDS.spkfParams = [alpha beta kappa];

% find UKF estimate for each satellite sensor given corrected LOS
% observation and Satellite positions, using fused estimate from earlier time step

[target.sen1_state_est(:,t), Px_1] = ukf(target.state_est(:,t-TS), Px, pNoise, oNoise,
Sensor(1).obs_ECIang_correct_EKF(:,k1), [], Sat(1).posit(:,t), InfDS);
[target.sen2_state_est(:,t), Px_2] = ukf(target.state_est(:,t-TS), Px, pNoise, oNoise,
Sensor(2).obs_ECIang_correct_EKF(:,k2), [], Sat(2).posit(:,t), InfDS);

end     % end case

% ---------------- fuse estimates for EKF and UKF tracking ----------------
if kk > 1
   Px_ = feval(model.prior, model, Px, pNoise);   % update state covariance
   Xh_ = feval(model.ffun, model, target.state_est(:,t-TS), pNoise.mu, []);  % project last estimate forward

   Px = inv(inv(Px_1) + inv(Px_2)-inv(Px_));       % fused covariance
   target.state_est(:,t) = Px*(inv(Px_1)*target.sen1_state_est(:,t) + inv(Px_2)*target.sen2_state_est(:,t)-
inv(Px_)*Xh_);  % new fused estimate
end  % if kk > 1

% ------------------------------------------------------------------------
end % end intrack

% figure(1);
% hold on;
% grid on;
% plot3(target.posit(1,t), target.posit(2,t), target.posit(3,t),'k*');
% plot3(target.sen1_state_est(1,t), target.sen1_state_est(3,t), target.sen1_state_est(5,t),'b*');
% plot3(target.sen2_state_est(1,t), target.sen2_state_est(3,t), target.sen2_state_est(5,t),'r*');
% plot3(target.state_est(1,t), target.state_est(3,t), target.state_est(5,t),'g*');
% drawnow
%  time = t

end     % end if t > newtrack

% -----------------------------------
% Filter consistency check
target.state_pk(nr,t) = sqrt(Px(1,1)^2 + Px(3,3)^2 + Px(5,5)^2);

est_error(:,t) = target.true_state(:,t) - target.state_est(:,t);
epsilon(nr,t) = est_error(:,t)' * inv(Px) * est_error(:,t);

% --------------------------------------------------------------------
end     % end t

avest = round((300+TS)/TS);
aveend = round((kend-kstart)/TS+1);

%  ---------------------- calculate errors ------------------------------
target.vel_est = sqrt(target.state_est(2,:).^2 + target.state_est(4,:).^2 + target.state_est(6,:).^2);  % velocity
error

% error.int_nocorr = error w/out bias correction or LKF
error(kk, b).int_nocorr(nr,:) = sqrt((target.intercept_nocorr(1,kstart:kend) - target.posit(1,kstart:kend)).^2 +
(target.intercept_nocorr(2,kstart:kend) - target.posit(2,kstart:kend)).^2 +
(target.intercept_nocorr(3,kstart:kend) - target.posit(3,kstart:kend)).^2);

% error.int_EKF = error w/bias correction EKF no LKF
error(kk, b).int_EKF(nr,:) = sqrt((target.intercept_EKF(1,kstart:kend) - target.posit(1,kstart:kend)).^2 +
(target.intercept_EKF(2,kstart:kend) - target.posit(2,kstart:kend)).^2 + (target.intercept_EKF(3,kstart:kend)
- target.posit(3,kstart:kend)).^2);

% error.int_sh = error w/ sample and hold bias correction EKF no LKF
error(kk, b).int_sh(nr,:) = sqrt((target.intercept_sh(1,kstart:kend) - target.posit(1,kstart:kend)).^2 +
(target.intercept_sh(2,kstart:kend) - target.posit(2,kstart:kend)).^2 + (target.intercept_sh(3,kstart:kend) -
target.posit(3,kstart:kend)).^2);

% error.state_est = error w/bias correction EKF and State Estimator
error(kk, b).state_est(nr,:) = sqrt((target.state_est(1,kstart:kend) - target.posit(1,kstart:kend)).^2 +
(target.state_est(3,kstart:kend) - target.posit(2,kstart:kend)).^2 + (target.state_est(5,kstart:kend) -
target.posit(3,kstart:kend)).^2);

% determine error for run nr over tracked time period t = 300 to Tplot
error(kk, b).int_nocorr_ave(nr) = mean(error(kk, b).int_nocorr(nr,avest:TS:kend));
error(kk, b).int_EKF_ave(nr) = mean(error(kk, b).int_EKF(nr,avest:TS:kend));
error(kk, b).int_sh_ave(nr) = mean(error(kk, b).int_sh(nr,avest:TS:kend));
error(kk, b).state_est_ave(nr) = mean(error(kk, b).state_est(nr,avest:TS:kend));
number = nr
end     % end Nruns

error(kk, b).state_pk_ave = sum(target.state_pk)/nr;

% average error over number of runs at each time step
error(kk, b).state_est_timeave = sum(error(kk, b).state_est)/nr;
error(kk, b).int_EKF_timeave = sum(error(kk, b).int_EKF)/nr;
error(kk, b).int_sh_timeave = sum(error(kk, b).int_sh)/nr;
error(kk, b).int_nocorr_timeave =sum(error(kk, b).int_nocorr)/nr;

error(kk, b).state_est_MSE = sum(error(kk, b).state_est.^2)/nr;
error(kk, b).int_EKF_MSE = sum(error(kk, b).int_EKF.^2)/nr;
error(kk, b).int_sh_MSE = sum(error(kk, b).int_sh.^2)/nr;
error(kk, b).int_nocorr_MSE = sum(error(kk, b).int_nocorr.^2)/nr;

error(kk, b).state_est_RMS = mean(sqrt(sum(error(kk, b).state_est(:,avest:TS:kend).^2)/nr));
error(kk, b).int_EKF_RMS = mean(sqrt(sum(error(kk, b).int_EKF(:,avest:TS:kend).^2)/nr));
error(kk, b).int_sh_RMS = mean(sqrt(sum(error(kk, b).int_sh(:,avest:TS:kend).^2)/nr));
error(kk, b).int_nocorr_RMS = mean(sqrt(sum(error(kk, b).int_nocorr(:,avest:TS:kend).^2)/nr));

error(kk, b).state_est_Mean_MSE = mean(error(kk, b).state_est_MSE(avest:TS:kend));
error(kk, b).int_EKF_Mean_MSE = mean(error(kk, b).int_EKF_MSE(avest:TS:kend));
error(kk, b).int_sh_MSE = mean(error(kk, b).int_sh_MSE(avest:TS:kend));
error(kk, b).int_nocorr_MSE = mean(error(kk, b).int_nocorr_MSE(avest:TS:kend));

if b==3
epsilon_ave(kk,:) = sum(epsilon)/nr;
figure(20)
hold on
plot(501:TS:kend, epsilon_ave(kk,501:TS:kend), lk)
xlabel('Time (seconds)');
ylabel('epsilon');
legend('LKF','EKF','UKF');
end

% for CRLB estimate, error standard deviation for time average
error(kk, b).state_est_STD = std(error(kk, b).state_est,0,1);

% determine mean of error over entire run
error(kk, b).state_est_runave = mean(error(kk, b).state_est_ave);
error(kk, b).int_EKF_runave = mean(error(kk, b).int_EKF_ave);
error(kk, b).int_sh_runave = mean(error(kk, b).int_sh_ave);
error(kk, b).int_nocorr_runave = mean(error(kk, b).int_nocorr_ave);

% determine Standard deviation of error over run
error(kk, b).state_est_runSTD = std(error(kk, b).state_est_ave);
error(kk, b).int_EKF_runSTD = std(error(kk, b).int_EKF_ave);
error(kk, b).int_sh_runSTD = std(error(kk, b).int_sh_ave);
error(kk, b).int_nocorr_runSTD = std(error(kk, b).int_nocorr_ave);

MSE(kk,b) = sqrt(error(kk, b).state_est_Mean_MSE);
RMS(kk,b) = error(kk, b).state_est_RMS;
Mean_error(kk,b) = error(kk, b).state_est_runave;

improv(kk, b).int_EKF = (1 - error(kk, b).int_EKF_runave/error(kk, b).int_nocorr_runave)*100;
improv(kk, b).int_sh =  (1 - error(kk, b).int_sh_runave/error(kk, b).int_nocorr_runave)*100;
improv(kk, b).state_est = (1 - error(kk, b).state_est_runave/error(kk, b).int_nocorr_runave)*100;

if  (b == 3) && (kk == 3)
%     % Plot average error
% figure(3);
% grid on
% p32 = plot(1:10:Tplot, 1000*error(kk, b).int_nocorr_timeave(1:10:Tplot), 'k-');hold on
% p33 = plot(1:10:Tplot, 1000*error(kk, b).int_EKF_timeave(1:10:Tplot), 'b-');
% p34 = plot(1:10:Tplot, 1000*error(kk, b).int_sh_timeave(1:10:Tplot), 'r-');
% % p35 = plot(1:10:Tplot, 1000*error(kk, b).state_est_timeave(1:10:Tplot), 'g-');
% title('Mean Error of target with and without bias correction');
% xlabel('Time (seconds)');
% ylabel('Mean Error (m)');
% legend('Without bias correction','Bias EKF correction','Bias sample and hold correction');  %, 'Bias EKF and State LKF'
%
figure(5);
hold on;
p55 = plot(1:Tplot, bsaz(b,1:Tplot), 'b-.');
P56 = plot(1:Tplot, bsel(b,1:Tplot), 'r-.');
p51 = plot(1:TS:Tplot, Sensor(1).bias_sh(2,1:TS:Tplot), 'k:');
p52 = plot(1:TS:Tplot, Sensor(1).bias_est(2,1:TS:Tplot), 'k-');
%  p53 = plot(1:TS:Tplot, Sensor(2).bias_sh(1,1:TS:Tplot), 'b:');
%  p54 = plot(1:TS:Tplot, Sensor(2).bias_sh(2,1:TS:Tplot), 'k:');
p57 = plot(1:TS:Tplot, Sensor(1).bias_sh(1,1:TS:Tplot), 'k:');
p58 = plot(1:TS:Tplot, Sensor(1).bias_est(1,1:TS:Tplot), 'k-');
xlabel('Time (secs)');
ylabel('Bias (rads)');
% legend('Without bias filter', 'With bias filter', 'True bias');
title('Sensor 1 Bias measurements without and with Kalman Filtering');

figure(4);
hold on;
p45 = plot(1:Tplot, bsaz(b,1:Tplot), 'b-.');
P46 = plot(1:Tplot, bsel(b,1:Tplot), 'r-.');
p41 = plot(1:TS:Tplot, Sensor(2).bias_sh(2,1:TS:Tplot), 'k:');
p42 = plot(1:TS:Tplot, Sensor(2).bias_est(2,1:TS:Tplot), 'k-');
%  p43 = plot(1:TS:Tplot, Sensor(2).bias_sh(1,1:TS:Tplot), 'b:');
%  p44 = plot(1:TS:Tplot, Sensor(2).bias_sh(2,1:TS:Tplot), 'k:');
p47 = plot(1:TS:Tplot, Sensor(2).bias_sh(1,1:TS:Tplot), 'k:');
p48 = plot(1:TS:Tplot, Sensor(2).bias_est(1,1:TS:Tplot), 'k-');
title('Sensor 2 Bias measurements without and with Kalman Filtering');
xlabel('Time (secs)');
ylabel('Bias (rads)');

end   % b==3

% Print results
results.error(b+1,1) = biasvect(b)';
results.error(b+1,2) = error(kk, b).int_nocorr_runave;
results.error(b+1,3) = error(kk, b).int_nocorr_RMS;
results.error(b+1,4) = error(kk, b).int_EKF_runave;
results.error(b+1,5) = error(kk, b).int_EKF_RMS;
results.error(b+1,6) = improv(kk, b).int_EKF;
results.error(b+1,7) = error(kk, b).int_sh_runave;
results.error(b+1,8) = error(kk, b).int_sh_RMS;
results.error(b+1,9) = improv(kk, b).int_sh;
results.error(b+1,10) = error(kk, b).state_est_runave;
results.error(b+1,11) = error(kk, b).state_est_RMS;
results.error(b+1,12) = improv(kk, b).state_est;
```

```
data = results.error(b+1,:)
stars = [nstars_1 nstars_2]
fprintf(savefile, '%12.6e\t', data);
fprintf(savefile, '\r Number of stars: Sensor 1: %6.0f\t, Sensor 2: %6.0f\n', stars);
fprintf(savefile, 'MSE: \t');
fprintf(savefile, '%12.6e\t', MSE(kk,:));
fprintf(savefile, 'RMS: \t');
fprintf(savefile, '%12.6e\t', RMS(kk,:));
fprintf(savefile, 'Mean Error: \t');
fprintf(savefile, '%12.6e\t', Mean_error(kk,:));

if b == 3
figure(6);
hold on;
p64 = plot(kstart:10:kend, error(kk, b).state_est_timeave(1:10:kend-kstart+1), lk);
title('Mean Error of target with EKF bias correction');
xlabel('Time (seconds)');
ylabel('Mean Error Km');
axis([0 1800 0 1]);
end

end     % end bias

end     % end kk

% ------------------------------ Calculate and plot PCRLB -----------------------

tcrb = 600;

Qn = [1/3 1/2 0 0 0 0;
      1/2 1   0 0 0 0;
      0 0    1/3 1/2 0 0;
      0 0    1/2 1   0 0;
      0 0 0 0       1/3 1/2;
      0 0 0 0       1/2  1];

  Qcrb = process_error*Qn/1000;

for t = 1:tcrb
Jz_1 = PCRLB(target.posit(:,t), Sat(1).posit(:,t), oNoise);
Jz_2 = PCRLB(target.posit(:,t), Sat(2).posit(:,t), oNoise);

if t == 1
    J(:,:,t) = eye(6);
else
J(:,:,t) = inv(Qcrb+ A*inv(J(:,:,t-1))*A') + .5*Jz_1 + .5*Jz_2;
end     % if t==0

CRLB(:,:,t) = inv(J(:,:,t));
CRLBave(t) = CRLB(1,1,t)^2 + CRLB(3,3,t)^2 + CRLB(5,5,t)^2;
end     % end t 1-50

figure(8)
```

```
hold on;
p81 = plot(101:10:tcrb, CRLBave(101:10:tcrb), 'k-');
p82 = plot(101:10:tcrb, error(1,3).state_est_STD(101:10:tcrb).^2, 'r-');
p83 = plot(101:10:tcrb, error(2,3).state_est_STD(101:10:tcrb).^2, 'b-');
p84 = plot(101:10:tcrb, error(3,3).state_est_STD(101:10:tcrb).^2, 'g-');
p85 = plot(101:10:tcrb, error(3,3).state_pk_ave(1,101:10:tcrb),'r-');
ylabel('Error variance (km)2');
xlabel('time (seconds)');
legend('PCRLB', 'LKF', 'EKF', 'UKF');

% ----------------------------------------------------------------------

figure(6);
hold on
p61 = plot(kstart:10:kend, error(2,3).int_nocorr_timeave(1:10:kend-kstart+1), 'k-');
legend('With LKF','With EKF', 'With UKF','Without bias correction');

% % plot observation vector trace
figure(2)
hold on
grid on
xlabel('Longitude')
ylabel('Latitude')
title('Sensor tracks and star detections');
% drawnow
p21 = plot((180/pi)*Sensor(1).obs_ECIae_noisy(1,300:Tplot),
(180/pi)*Sensor(1).obs_ECIae_noisy(2,300:Tplot), 'r:');
p22 = plot((180/pi)*Sensor(2).obs_ECIae_noisy(1,300:Tplot),
(180/pi)*Sensor(2).obs_ECIae_noisy(2,300:Tplot), 'b-') ;
p23 = plot((180/pi)*starplot1(1,:), (180/pi)*starplot1(2,:), 'k*');
p24 = plot((180/pi)*starplot2(1,:), (180/pi)*starplot2(2,:), 'k*');
axis([0 360 -90 90])
legend('Sensor 1 track', 'Sensor 2 track', 'Detected star');

% fprintf(savefile, 'MSE: %12.6e\n', MSE);
% fprintf(savefile, 'RMS: %12.6e\n', RMS);
% fprintf(savefile, 'Mean_error: %12.6e\n', Mean_error);


fclose(savefile)
```

Bias Extended Kalman Filter

```
function [xBhnext, PBnext] = Bias_KF(dt, xBh, PB, RB, ZB, QB)
% produces bias state estimate and covariance matrix based on bias
% measurement over the time step dt.
% inputs
% dt = time step
% xBh = previous state
% PB = previous state cov matrix
% ZB = bias measurement
% RB = measurement error variance
% QB = bias process variance
% Outputs
% xBhnext = Bias state estimate
% PBnext = bias state cov matrix

wh = xBh(3);
xh = xBh(1);

HB = [1 0 0];  % measurement matrix

for t = 1:dt;

  THB = [1  dt  0;
        -wh*wh*dt  1   -2*wh*xh*dt;
        0   1   0];

  GB = [ 0   0   0;
         0   4*wh*wh*xh*xh*dt*dt*dt/3   -2*wh*xh*dt*dt;
         0   -2*wh*xh*dt*dt       dt];

 % Ricati equations

  M = THB*PB*THB' + GB*QB;
  K = M*HB'*inv(HB*M*HB' + RB);          % Kalman gain
  PBnext = (eye(3)-K*HB)*M;              % update state covariance
  xBh_ = project2(dt, xBh, 100);      % numerical propagate state vector
  inov = ZB-HB*xBh_;
  xBhnext = xBh_ + K*inov;               % update state

    end
```

Quateron transfer function

```
function B = quatxfrm(q, A)

% Quaternion transform
% This program provides quaternion conversion.
% Given a matrix of vectors, A in 3 dimensions, and the attitude quaternons,
% q and returns the output vector, B in the new coordinate frame

nov = size(q,2);
B = zeros(3,nov);
D = zeros(3,3,nov);
D(1,1,:) = 2*(q(1,:).^2 + q(4,:).^2 - 0.5);
D(1,2,:) = 2*(q(1,:).*q(2,:) + q(3,:).*q(4,:));
D(1,3,:) = 2*(q(1,:).*q(3,:) - q(2,:).*q(4,:));
D(2,1,:) = 2*(q(1,:).*q(2,:) - q(3,:).*q(4,:));
D(2,2,:) = 2*(q(2,:).^2 + q(4,:).^2 - 0.5);
D(2,3,:) = 2*(q(2,:).*q(3,:) + q(1,:).*q(4,:));
D(3,1,:) = 2*(q(1,:).*q(3,:) + q(2,:).*q(4,:));
D(3,2,:) = 2*(q(2,:).*q(3,:) - q(1,:).*q(4,:));
D(3,3,:) = 2*(q(3,:).^2 + q(4,:).^2 - 0.5);

for i = 1:nov
  B(:,i) = D(:,:,i)^-1*A(:,i);
end
```

Posterior Cramer-Rao Lower Bound Function

```
function Jz = PCRLB(target, Sat, oNoise)

R = oNoise.cov;
r = sqrt((target(1) - Sat(1))^2 + (target(2) - Sat(2))^2 + (target(3) - Sat(3))^2);

H(1,1) = -(target(2) - Sat(2))/((target(1) - Sat(1))^2 + (target(2) - Sat(2))^2);
H(1,3) = (target(1) - Sat(1))/((target(1) - Sat(1))^2 + (target(2) - Sat(2))^2);
H(1,5) = 0;
H(1,2) = 0;
H(1,4) = 0;
H(1,6) = 0;
H(2,1) = -((target(1) - Sat(1))*(target(3) - Sat(3)))/(r^2 * sqrt(((target(1) - Sat(1))^2 + (target(2) -
Sat(2))^2)));
H(2,3) = -((target(2) - Sat(2))*(target(3) - Sat(3)))/(r^2 * sqrt(((target(1) - Sat(1))^2 + (target(2) -
Sat(2))^2)));
H(2,5) = sqrt((target(1) - Sat(1))^2 + (target(2) - Sat(2))^2)/r^2;
H(2,2) = 0;
H(2,4) = 0;
H(2,6) = 0;

Jz = H'*inv(R)*H;
```

Linearized Kalman Filter Function

```
function [Xh, Px] = LKFts(model, pNoise, state, P, covsen, obs_1, obs_2, sat_1, sat_2, intercept_)
global TS
% inputs
% model =
% pNoise = process noise covariance
% state = target state estimate at time t-1
% P = state covariance at t-1
% covsen = sensor covariance
% obs_1 = sensor 1 observation at time t
% obs_2 = sensor 2 observation at time t
% sat_1 = satellite 1 position
% sat_2 = satellite 2 position
% intercept_ = the intercept point of the observation vectors in ECI

% outputs
% Xh = state estimate at time t
% Px = state estimate covariance at time t


  A = model.A;
  G = model.G;
  B = model.B;

  H = [1 0 0 0 0 0;      % observation vector
     0 0 1 0 0 0;
     0 0 0 0 1 0];

 %-----------------------------------------------------
  %--- TIME UPDATE
  alt = sqrt(state(1)^2 + state(3)^2 + state(5)^2);    % target altitude

  Xh_ = feval(model.ffun, model, state, pNoise.mu, []);
  Px_     = A*P*A' + G*pNoise.cov*G';

  %--- MEASUREMENT UPDATE
  % range from satellite 1 to target positiion estimate
  a = sqrt((Xh_(1)-sat_1(1)).^2 + (Xh_(3)-sat_1(2)).^2 + (Xh_(5)-sat_1(3)).^2);
  % range from satellite 2 to target positiion estimate
  a2 = sqrt((Xh_(1)-sat_2(1)).^2 + (Xh_(3)-sat_2(2)).^2 + (Xh_(5)-sat_2(3)).^2);


  alpha = cos(dot(obs_1,obs_2));  % cross angle between observation vectors

  if alpha > 2*pi/4
    alpha = alpha - 2*pi/4;
       if alpha > pi/4
          alpha = alpha - pi/4;
       end
  end
```

```
   covrange = a2*covsen/cos(alpha/2);      % range covariance based on satellite cross angle

   % Adjust for conversion from polar to vector measurements

   Tm = [cos(obs_1(2)).*cos(obs_1(1))  (-a)*cos(obs_1(2)).*sin(obs_1(1)) (-a) *…
sin(obs_1(2)) .*cos(obs_1(1));
     sin(obs_1(2)).*cos(obs_1(1))       a*cos(obs_1(2)).*cos(obs_1(1)) (-a)*sin(obs_1(2)).*sin(obs_1(1));
     sin(obs_1(2))                    0                          a*cos(obs_1(2))];

   C = [covrange^2 0 0;        % Cartesian error matrix
       0   covsen^2 0;
       0   0 covsen^2];

   R1 = Tm*C*Tm' ;

   Py        = H*Px_*H' + R1;
   KG        = Px_ * H' * inv(Py);

  % calculate projected azimuth and elevation angles
  y1_(1) = azimuth(Xh_(1)-sat_1(1), Xh_(3)-sat_1(2));
  y1_(2) = atan2(Xh_(5)-sat_1(3),sqrt((Xh_(1)-sat_1(1)).^2 + (Xh_(3)-sat_1(2)).^2));
  y2_(1) = azimuth(Xh_(1)-sat_2(1), Xh_(3)-sat_2(2));
  y2_(2) = atan2(Xh_(5)-sat_2(3),sqrt((Xh_(1)-sat_2(1)).^2 + (Xh_(3)-sat_2(2)).^2));

  % convert angles to vectors
   y1_v = brngtovec(y1_(:));
   y2_v = brngtovec(y2_(:));

  % calculate projected intercept point (this becomes projected observation)
   yh_ = intercept(y1_v, y2_v, sat_1, sat_2);

   inov = intercept_ - yh_ - H*(9.6e-3)*B*state/alt ;  % observsation inovation

   Xh   = Xh_ + KG * inov;   % new LKF state estimate
   Px   = Px_ - KG*Py*KG';   % new LKF state covariance matrix
```

APPENDIX C: ORBITOLOGY BASICS

The four basic orbits associated with space based tracking systems are described here and shown in Figure C-1.

Low Earth Orbit (LEO): The primary orbit of the space based sensor satellites is the Low Earth Orbit (LEO). This orbit is in the 100 – 1000 mile altitude range. Satellites in this orbit move rapidly with respect to the earth's surface and complete an orbit in a period of 90-120 minutes.

Ballistic Flight: The next significant orbit associated with this study is the ballistic flight. In reality a ballistic missile is a low earth orbit satellite where the perigee (lowest point of approach) of the orbit is less than the earth's radius.

Geostationary Orbit (GEO): In the GEO orbit the satellite travels in a manner that it maintains a constant position above the earth. Communications satellites and the missile launch detection satellites are in this orbit.

Highly Elliptical Orbit (HEO): HEO orbits are designed to allow the satellite to dwell for 8 hours over the northern hemisphere and are typically at a 64 degree inclination.

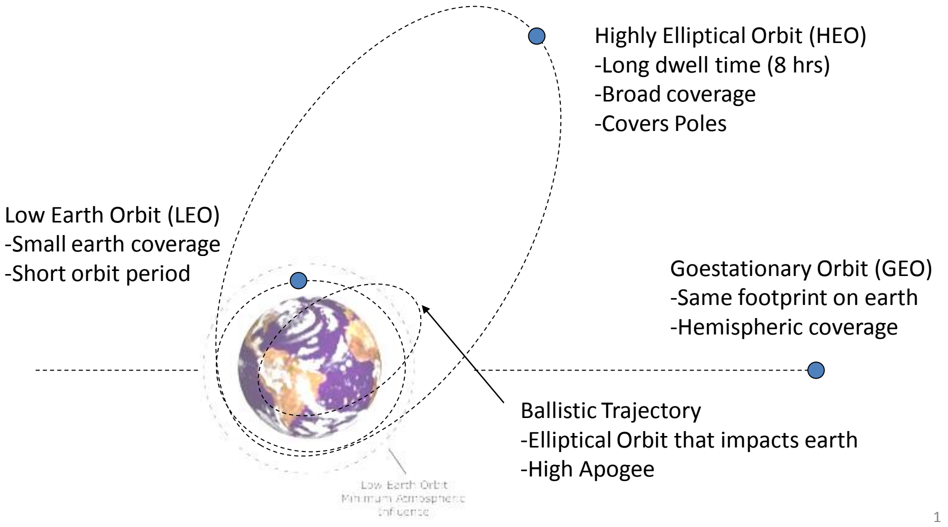


Figure C-1. Basic orbits

The phases of ballistic missile flight are shown in Figure C-2 and described here.

Boost phase: Boost phase occurs from the time the missile is launched until the booster rocket ceases to burn. The velocity vector at burnout determines the direction and range of the missile.

Midcourse phase: In this phase of flight the missile trajectory is determined by the effects of gravity. During this phase the missile follows an elliptical path around the earth’s center of gravity. Also during this phase, the warhead will separate from the final booster stage. It is at this point decoys will be deployed. The booster is typically accompanied by the debris from the rest of the rocket, such as retaining rings, final booster stage, and warhead fairings.

Terminal Phase: Once the ballistic missile begins to reenter the atmosphere it enters the terminal phase of flight. In this phase the lighter, less aerodynamic debris and decoys will begin to strip away from the warhead. The deceleration of the warhead during reentry depends upon the ballistic coefficient and is proportional to the density of the atmosphere.

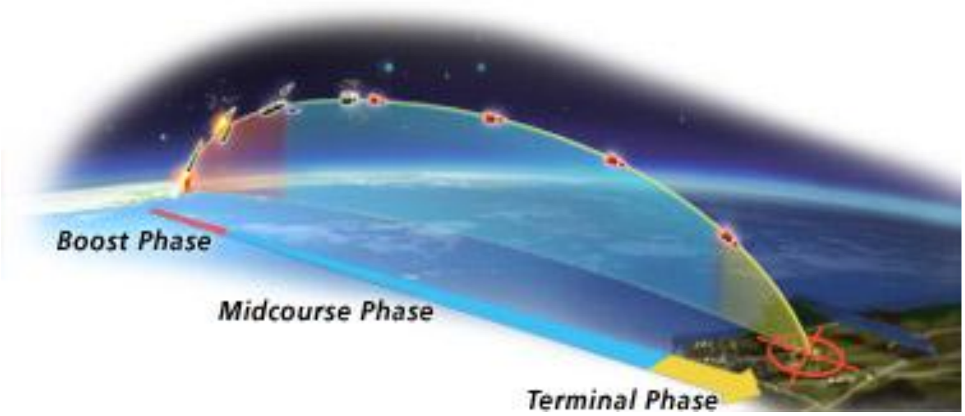


Figure C-2. Phases of Ballistic Missile Flight

REFERENCES

REFERENCES

[1] Watson, John, and Keith Zondervan. *The Missile Defense Agency's Space Tracking and Surveillance System.* Web Spring 2008. <http://www.aero.org/>

[2] *Space Tracking and Surveillance System.* MDA Factsheet 03-SF-0013, Web Fall 2003. <www.mda.mil>

[3] Vallado, David A. *Fundamentals of Astrodynamics and Applications*. New York: McGraw Hill, 1997.

[4] Minvielle, Pierre. "Decades of Improvements in Re-entry Vehicle Tracking." *IEEE Aerospace and Electronic Systems Magazine* 20.8 (2005): CF1-14.

[5] Blackman, Samuel S., and Robert Popoli. *Design and Analysis of Modern Tracking Systems*. Norwood, MA: Artech House, 1999.

[6] Kalman, Rudolf E. "New Results in Linear Filtering and Prediction Theory." *Journal of Basic Engineering* 83 (1961): 95-108.

[7] Farina, Alfonso, Branko Ristic, and Dario Benvenuti. "Tracking a Ballistic Target: Comparison of Several Nonlinear Filters." *IEEE Trans. on Aerospace and Electronic Systems* 38.3 (2002): 854-66.

[8] Bar-Shalom, Yaakov, X. Rong Li, and Thiagalingam Kirubarajan. *Estimation with Applications to Tracking and Navigation*. New York: Wiley-Interscience, 2001.

[9] Bar-Shalom, Yaakov and X. Rong Li. *Estimation and Tracking – Principles, Techniques and Software*. Norwood, MA: Artech House, 1993.

[10] Gleb, Arthur. *Applied Optimal Estimation*. Cambridge, MA: The MIT Press, 1988.

[11] Van der Merwe, R., N. Freitas, A. Doucet, and E. Wan. The Unscented Particle Filter. Tech. no. CUED/F-INFENG/TR 380. Cambridge University Engineering Department, 2000.

[12] Sauson, Brian G., and Kuo-Chu Chang. "Nonlinear Estimation Comparison for Ballistic Missile Tracking." *Optical Engineering* 43.6 (2004): 1-14.

[13] Chang, C. B. "Ballistic Trajectory Estimation with Angle-only Measurements." *IEEE Trans. on Automatic Control* 25.3 (1980): 474-80.

[14] Yeddanapudi, Murali, Yaakov Bar-Shalom, Krishna R. Pattipati, and Somnath Deb. "Ballistic Missile Track Initiation from Satellite Observations." *IEEE Trans. on Aerospace and Electronic Systems* 31.3 (1995): 1054-071.

[15] Aidala, Vincent J. "Kalman Filter Behavior in Bearings-only Tracking Applications." *IEEE Trans. on Aerospace and Electronic Systems* 15.1 (1979): 29-39.

[16] Alfonso Farina. "Target Tracking With Bearings-only Measurements." *Signal Processing* 78 (1999): 61-78.

[17] Roa, S. Koteswara. "Algorithm for Detection of Maneuvering Targets in Bearings-only Passive Target Tracking." *IEEE Proc. on Radar, Sonar and Navigation* 146.3 (1999): 141-46.

[18] Roa, S. Koteswara. "Pseudo-linear Estimator for Bearings-only Passive Target Tracking." *IEEE Proc. on Radar, Sonar, and Navigation* 148.1 (2001): 16-22.

[19] Haessig, David A., and Bernard Friedland. "Maximum Likelihood Estimation of Target Acceleration." *Proc. of 23rd Conference on Decision and Control*. Las Vegas, NV, (1984): 1398-402.

[20] Lindgren, Allen G., and Kai F. Gong. "Position and Velocity Estimation via Bearing Observations." *IEEE Trans. on Aerospace and Electronic Systems* 14.4 (1978): 564-76.

[21] Friedland, Bernard. “Treatment of Bias in Recursive Filtering,” *IEEE Trans. on Automatic Control* 14.4 (1969): 359-67.

[22] Friedland, Bernard. “Notes on Separate-Bias Estimation,” *IEEE Trans. on Automatic Control* 23.4 (1978): 735-38.

[23] Ignagni, M. B. “An Alternate Derivation and Extension of Friendland’s Two-stage Kalman estimator,” *IEEE Trans. on Automatic Control* 26.3 (1981): 746-50.

[24] Ignagni, M. B. “Separate-Bias Kalman Estimator with Bias State Noise,” *IEEE Trans. on Automatic Control* 35.3 (1990): 338-41.

[25] Haessig, David, and Bernard Friedland. "Separate-Bias Estimation with Reduced-Order Kalman Filters." *IEEE Trans. on Automatic Control* 43.7 (1998): 983-87.

[26] Kastella, K., B. Yeary, T. Zadra, R. Brouilard, and E. Frangione. "Bias Modeling and Estimation for GMTI Applications." *Proc. of 3rd Int. Conf. on Information Fusion, (FUSION2000).* Vol. 1. 2000. TuC1/7-uC1/12.

[27] Bar-Shalom, Yaakov. "Airborne GMTI Radar Position Bias Estimation Using Static-Rotator Targets of Opportunity." *IEEE Trans. on Aerospace and Electronic Systems* 29.4 (2001): 695-98.

[28] Helmick, R. E., and T. R. Rice. “Removal of Alignment Errors in an Integrated System of Two 3-D Sensors,” *IEEE Trans. on Aerospace and Electronic Systems* 29.4 (1993): 1333-343.

[29] Kosuge, Y., and T. Okada. "Statistical Analysis for Radar Bias Error Estimation in a Data Fusion System of 3-Dimensional Radars." *Proc. of 26th Annual Conference of the IEEE Industrial Electronics Society, (IECON2000)*. Vol. 3. 2001-006.

[30] Osborne III, Richard., Yaakov Bar-Shalom and T. Kirubarajan. "Radar Measurement Noise Variance Estimation with Several Targets of Opportunity." *IEEE Trans. on Aerospace and Electronic Systems* 44.3 (2008): 985-95.

[31] Kistosturian, Haig G., "On-orbit Calibration of Satellite Antenna Pointing Errors." *IEEE Trans. on Aerospace and Electronic Systems* 26.1 (1990): 88-112.

[32] Bate, Roger R. *Fundamentals of Astrodynamics*. New York: Dover Publications, 1971.

[33] Xing, Fei, Ying Dong, Zheng You, and Qin Zhou. “APS Star Tracker and Its Attitude Estimation,” *Proc. of 1st Int. Symposium on Systems and Control in Aerospace and Aeronautics*. 2006. 34-38.

[34] "STK Product Suite." *Analytical Graphics, Inc. (AGI), analysis software for land, sea, air, and space*. Web. Winter 2010. <http://www.agi.com>.

[35] Clemons III, Thomas M., and Kuo-Chu Chang. “Effect of Sensor Bias on Space-based Bearings-only tracker.” *Proc. of SPIE,* Vol. 6968. (2008): 696807.1-12.

[36] Gray, Roger L. “A Pure-Cartesian Formulation for Tracking Filters,” *IEEE Trans. on Aerospace and Electronic Systems* 29.3 (1993): 749-54.

[37] Liebe, Carl C. "Accuracy Performance of Star Trackers - A Tutorial." *IEEE Trans. on Aerospace and Electronic Systems* 38.2 (2002).

[38] Li, Hui, Zhao-wei Sum, and Shi-jie Zhang. “Satellite Attitude Estimation UKF Algorithm Based on Star-Sensor,” *Proc. of IMACS Multiconfernce on “Computational Engineering in Systems Applications” (CESA),* Beijing, China, (2006): 799-802.

[39] Wu, Andy, “SBIRS High Payload LOS Attitude Determination and Calibration,” *Proc of IEEE Aerospace Conference* Vol. 5 (1998): 243 – 53.

[40] Sande, C., and N. Ottenstein. *SKYMMAP Requirements, Functions, and Mathematical Specifications*. Tech. no. CSC-96-932-24. 3rd rev. Vol. 3. Greenbelt, MD: Computer Sciences Corporation, 1999.

[41] Clemons III, Thomas M., and Kuo-Chu Chang. “Space-based IR tracking bias removal using background star observations.” *Proc. of SPIE*. Vol. 7330. (2009): 733013.1-12.

[42] Clemons III, Thomas M., and Kuo-Chu Chang. “Bias Correction Using Background stars for Space Based IR Tracking.” *Proceedings of the 12th International Conference on Information Fusion (FUSION 09),* (2009): 2028 – 2035.

[43] Samann, Malak A., Daniele Mortari, and John L. Junkins. “Recursive Mode Star Identification Algorithms.” *IEEE Trans. on Aerospace and Electronic Systems* 41.4 (2005): 1246-54.

[44] Zarchan, Paul. *Fundamentals of Kalman Filtering a Practical Approach*. Reston, Va: American Institute of Aeronautics and Astronautics, 2000.

[45] Van der Merwe, Rodoplph, and Eric Wan. *Recursive Bayesian Estimation Laboratory Toolkit*. Computer software. *REBEL: Recursive Bayesian Estimation Laboratory*. Vers. 0.2.7. Oregon Health & Science University, 11 Oct. 2006. Web. Autumn 2007. <http://choosh.csee.ogi.edu/rebel/>.

[46] Van Trees, H. *Detection, Estimation, and Modulation Theory*. New York: Wiley and Sons, 1968.

[47] Ristic, Bernard, Alfonso Farina, and Marcel L. Hernandez. “Cramer-Rao Lower Bound for Tracking Multiple Targets.” *IEEE Proc. of Radar and Sonar Navigation* 151.3 (2004): 129-34.

[48] Hernandez, Marcel L., Alfonso Farina, and Bernard Ristic. “PCRLB for Tracking in Cluttered Environments: Measurement Sequence Conditioning Approach.” *IEEE Trans. on Aerospace and Electronic Systems* 42.2 (2006): 680-704.

[49] Dogancay, Kutluyil. “Bias Compensation for the Bearings-Only Pseudolinear Target Track Estimator.” *IEEE Trans. on Signal Processing* 54.1 (2006): 59-68.

[50] Ferry, James P. “Exact Bias Removal for the Track-to-Track Association Problem.” *Proc. of 12th Inter Conf on Information Fusion (Fusion 2009)* 2009: 1642-649.

[51] Chong, Chee-Yee. Kuo-Chu Chang, and S. Mori. "Distributed Multitarget Multisensor Tracking." Ed. Yaakov Bar-Shalom. *Multitarget-Multisensor Tracking: Advanced Applications*. Norwood, NH: Artech House, 1990. 247-95.

[52] Pyung Soo Kim. “Separate-Bias Estimation Scheme with Diversely Behaved Biases.” *IEEE Trans. on Aerospace and Electronic Systems* 38.1, (2002): 333-39.

[53] Sviestins, Egils. “On-line Bias Estimation for Multisensor Tracking.” *Proc. of Information, Decision and Control, 1999. (IDC 99)*. 1999: 221-226.

Aidala, Vincent J. and Nardone, S.C. “Biased Estimation Properties of the Pseudo Linear Tracking Filter,” *IEEE Trans. on Aerospace and Electronic Systems* 18.4 (1982): 432-41.

Aidala, Vincent J., and Hammel, S. E. “Utilization of Modified Polar Coordinates for Bearings-Only Tracking.” *IEEE Trans. on Automatic Control,* 28.3 (1983): 283-94.

Blackman, Samuel, and Robert Popoli. *Design and Analysis of Modern Tracking Systems (Artech House Radar Library)*. New York: Artech House, 1999.

Bruno, Marcelo, and Anton Pavlov. “Improved Particle Filters for Ballistic Missile Target Tracking.” *Proc. on IEEE International Conference on Acoustics, Speech, and Signal Processing, 2004 (ICASSP '04).* Vol. 2. (2004): 705-8.

Chang, Kuo-Chu and Chee-Yee Chong. "Distributed Estimation in Distributed-Sensor Networks." Ed. Yaakov Bar-Shalom. *Multitarget-Multisensor Tracking: Advanced Applications*. Norwood, NH: Artech House, 1990. 23-71.

Chen, Hiimin, Genshe Chen, Erik Blasch, and Khanh Pham. “Comparison of Several Space Target Tracking Filters.” *Proc. of SPIE*. Vol. 7330. (2009): 733001.1-12.

Chong, Chee-Yee. “Distributed Architectures for Data Fusion.” Proc. of International Conference on Sensor Fusion, 1998 (FUSION98). (1998): 84-91.

Clemons III, Thomas M., and Kuo-Chu Chang. “Estimation Filters for Missile Tracking with Airborne Laser.” *Proc of SPIE,* Vol. 6238. (2006): 623804.1-11.

Crassidis, John L. and F. Landis Markley. “Unscented Filtering for Spacecraft Attitude Estimation.” *Journal of Guidance Control and Dynamics,* 26.4 (2003): 536-42.

Crassidis, John L., F. Landis Markley and Yang Cheng. “Survey of Nonlinear Attitude Estimation Methods.” *Journal of Guidance Control and Dynamics*, 30.1, (2007): 12-28.

Danis, Norman J. “Space-Based Tactical Ballistic Missile Launch Parameter Estimation.” *IEEE Trans on Aerospace and Electronic Systems,* 29.2 (1993): 412-24.

Dikic, Goran, and Branko Kovacevic. “Target Tracking with Passive IR Sensors.” *Proc of 5th International Conference on Telecommunications in Modern Satellite, Cable and Broadcasting Service, 2001. TELSIKS 2001.* Vol. 2. (2001):745 – 48.

Hue, Carine, Jean-Pierre le Cadre, and Patrick Perez. "Posterior Cramer-Rao Bounds for Multi-Target Tracking." *IEEE Trans. on Aerospace and Electronic Systems* 42.1 (2006): 37-49.

Novoselov, Roman Y., Shawn M. Herman, Sabino M. Gadaleta, and Aubrey B. Poore. "Mitigating the Effects of Residual Biases with Schmidt-Kalman Filtering." *Proc. of 7th International Conference on Information Fusion, (FUSION2005).* (2005):358-65.

Psiaki, Mark L. "Attitude-Determination Filtering via Extended Quaternion Estimation." *Journal of Guidance Control and Dynamics,* 23.2 (2000): 206-214.

Reid, Donald B. "An Algorithm for Tracking Multiple Targets." *IEEE Trans. on Automatic Control* 24.6 (1979): 843-54.

Rhoads, Ian B. "A tutorial Introduction to Estimation and Filtering." *IEEE Trans. on Automatic Control* 16.6 (1971): 688-705.

Tichavsky, Petr, Carlos H. Muravchik, and Arye Nehorai. "Posterior Cramer-Rao Bounds for Discrete-Time Nonlinear Filtering." *IEEE Trans. On Signal Processing* 46.5 (1998): 1386-396.

Xiong, K., H. Y. Zhang, and C. W. Chan. "Performance Evaluation of UKF-based Nonlinear Filtering." *Automatica* 42 (2006): 261-70.

Xu, Ben-Lian, Qing-Lan Chen, Zheng-Yi Wu, Zhi-Quan Wang. "Analyis and Approximation of Performance Bound for Two-Observer Bearings Only Tracking." *Journal of Information Sciences* 178 (2008): 2059-2078.

CURRICULUM VITAE

Thomas M. Clemons III was born on August 12, 1960, and grew up in Marine City, MI, where he graduated from Marine City High School, in 1978. He then attended the United States Naval Academy and received his Bachelor of Science in Electrical Engineering in May 1982 and was commissioned and Ensign in the United States Navy. He served as a career Surface Warfare Officer specializing in Operations and Engineering. His duty included tours in four ships including Command of three Mine Countermeasures Ships, and deployed with his crew to the Persian Gulf. Following sea duty Tom served as a Program Integrator at the Ballistic Missile Defense Agency and as a Senior Analyst with the Joint Advanced Warfighting Program at the Institute for Defense Analyses, Alexandria, VA. Following retirement from the Navy at the rank of Commander, Tom was employed as a Senior Engineer with Dynetics Corporation of Huntsville, AL. In January 2004 he entered Federal Civil Service with the Department of Defense as a Senior Engineer in the Joint Chiefs of Staff, Director for Operations, Space and Missile Defense Division. Tom also holds a Masters of Science in Electrical Engineering (Space Systems Engineering) and the Degree of Electrical Engineer from the Naval Postgraduate School, Monterey, CA, December 1989, and a Masters of Arts (National Security and Strategic Studies) from the Naval War College, Newport, RI, June 1994.