The Pennsylvania State University
The Graduate School

COUPLING BETWEEN NONLINEAR ESTIMATION AND DYNAMIC SENSOR TASKING APPLIED TO SATELLITE TRACKING

A Dissertation in
Aerospace Engineering
by
Patrick S. Williams

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

December 2012

The dissertation of Patrick S. Williams was reviewed and approved* by the following:

David B. Spencer
Professor of Aerospace Engineering
Dissertation Advisor, Chair of Committee

Robert G. Melton
Professor of Aerospace Engineering

Jacob W. Langelaan
Associate Professor of Aerospace Engineering

Joseph P. Cusumano
Professor of Engineering Science and Mechanics

Richard S. Erwin
Principal Aerospace Engineer, Air Force Research Laboratory

George A. Lesieutre
Professor of Aerospace Engineering
Aerospace Engineering Department Head

*Signatures are on file in the Graduate School.

Abstract

The tracking of Earth orbiting objects has been a topic of growing concern, due to the fact that the amount of man-made orbital debris, and the number of active and inactive space objects have been steadily increasing over the past several decades. Space Situational Awareness (SSA) is concerned with the tracking, detection, and cataloging of numerous space objects using relatively few ground-and space-based sensors known as the Space Surveillance Network (SSN). While these sensors provide observations of object characteristics (range, azimuth, elevation, etc), the large number of objects compared to the limited sensors available to track them results in measurements occurring infrequently. These potentially long periods of either inability to make observations (due to line-of-sight access) or unavailability of sensors (due to scheduling constraints) necessitates the need to intelligently determine which objects should be observed and which should be ignored at various times, a process known as sensor tasking or sensor network management.

In order to make these tasking decisions, it is necessary to create some form of utility metric to determine which sensors should observe which objects at a particular instant of time. This dissertation examines the use of utility metrics from two forms of expected information gain for each object-sensor pair as well as the approximated stability of the estimation errors in order to work towards a tasking strategy. The information theoretic approaches use the calculation of Fisher information gain (FIG), an estimate of the upper bound of information present in an unbiased estimator, and Shannon information gain (SIG), a measure of information gained about the particular state. Both of these methods are considered myopic or greedy in nature, due to the fact that they calculate only information gained over one simulation time step. FIG has been studied previously as a potential sensor tasking metric, and has even been investigated in applications to SSA, while SIG has been suggested as a possible sensor tasking metric, but has yet to be investi-

gated when applied to sensor tasking in the SSA problem. The stability approach reflects a new type of metric referred to in these studies as largest Lyapunov exponent estimation (LLE), and has yet to be studied as a sensor tasking utility metric.

The process of evaluating these utility metrics is intrinsically tied in with state and uncertainty estimates provided by a nonlinear filter. That is, each utility metric requires estimates provided by the filters in order to be calculated, creating a coupling effect between the estimation and tasking components of the satellite tracking problem. In order to investigate this, three candidate nonlinear estimators, an extended Kalman filter (EKF), an unscented Kalman filter (UKF) and a recently introduced adaptive entropy-based Gaussian-mixture information synthesis (AEGIS) filter are tested. The primary difference in these filters is their ability to approximate system nonlinearities in their application, with previous work showing that the AEGIS filter performs the best in this regard, while the EKF performs the worst. While many studies have shown how an EKF and UKF differ in estimation performance when applied to orbit determination problems, little work has been done to investigate the AEGIS filter in these regards.

While much recent research has been conducted investigating specific methods of either sensor tasking or nonlinear estimation, there is yet to be any studies which investigate the coupling of the two, as it is related to overall tracking performance. The investigation of this coupling demonstrates that the use of more accurate filters leads to better overall estimates, not only due to the advantages within the estimation methods, but also from the improvement in tasking decisions due to selection of these estimators.

Table of Contents

List of Figures

List of Tables

List of Symbols

System Dynamics and Parameters

t	Time
t_f	Total simulation time
Δt	Simulation time step duration
N_s	Total space objects in simulation
$\mathcal{O}$	Set of object indicies
M_s	Total sensors in simulation
$\mathcal{S}$	Set of sensor indicies
$\vec{x}$	Object state vector
$\vec{w}$	Process noise vector
$\vec{s}$	Sensor state vector
n	Size of object/sensor state vectors
Φ	State transition matrix
Γ	Sensor field of regard
$\vec{y}$	Sensor measurement vector
ρ	Sensor range measurement
Δ	Maximum sensor range

ψ Sensor angle measurement

Ψ Maximum sensor half-angle

$\vec{v}$ Sensor noise vector

Estimation

$\hat{X}$ Estimated object state vector

$\hat{P}$ Estimated object covariance matrix

$\hat{\sigma}$ Estimated standard deviation

Q Process noise covariance matrix

$\hat{y}$ Estimated measurement

H Measurement function Jacobian

P^{xy} Cross covariance matrix

P^{yy} Innovation covariance matrix

R Sensor noise covariance matrix

K Kalman Gain matrix

$\mathcal{X}$ Sigma points

W_x^γ Sigma point weights for state estimate

W_p^γ Sigma point weights for covariance estimate

L Total components in Gaussian mixture model

ν Gaussian mixture model component weight

$\mathcal{H}$ Shannon entropy

Sensor Tasking

$\underline{\mu}$ Visibility matrix

$\underline{\xi}$ Decision matrix

Σ Fisher information matrix

Ω Fisher information gain matrix

ϕ Utility metric for Fisher information gain

I Shannon information

ΔI Utility metric for Shannon information gain

$\hat{\lambda}^1$ Utility metric for largest Lyapunov exponent estimation

Sumilation Performance and Parameters

Δr Difference in position between estimation and true value

$\langle E^r \rangle$ Average position error

$\langle \hat{A}_{err} \rangle$ Average estimated error ellipse area

$J^{\hat{P}}$ Multiplicative factor between true position error and estimated position standard deviation

Subscripts

k Simulation time step

i Object index

j Sensor index

Superscripts

a Indicates set of objects/sensos available for observation

τ Indicates set of objects/sensors tasked for observation

f Forecast estimate

$*$ Optimal estimate

γ Sigma point index

l Gaussian mixture model component

Functions and Operators

f Object dynamics

$\mathcal{F}$	Propagation of object dynamics
g_o	Orbiting sensor dynamics
g_g	Ground-based sensor dynamics
h	Sensor measurement function
$\mathcal{G}$	Propagation of sensor dynamics
p	Probability density function
p^g	Gaussian probability density function
p^{gmm}	Gaussian mixture model probability density function
$\mathbb{E}$	Expected value
∇	Differential

Acknowledgments

I would like to thank several people and organizations who helped me in this study. First, I would like to thank my advisor, Dr. David Spencer, for his continuous support in this research, and its funding.

I would also like to thank the Air Force Research Laboratory (AFRL) Space Vehicles Directorate for providing the opportunity to initiate research for this dissertation topic. Their funding, employees, and other resources available during the academic year and two summer internships helped mightily with the completion of this work. In particular, I would like to thank Dr. Scott Erwin for his mentorship and help in organizing the funding mechanisms for this research. Additionally, I would also like to thank AFRL employees Dr. Moriba Jah and Dr. Kyle DeMars for their assistance in introducing me to the adaptive entropy-based Gaussian information synthesis filter.

I would like to thank the additional members (besides Dr. David Spencer and Dr. Scott Erwin) my Ph.D. committee, Dr. Robert Melton, Dr. Jack Langelaan, Dr. Joseph Cusumano, and Karl Reichard for their advice and aid in this work.

I would like to thank the Pennsylvania State University for providing classes and resources which aided in these studies.

Finally, I would like to thank my family and friends, for which their continuous support was greatly appreciated.

Dedication

To Jo

Chapter 1

Introduction

1.1 Motivation

Beginning with the inception of the space program, the ability to accurately monitor man-made resident space objects has become of paramount importance to the continuation and safety of our near-Earth space operations. This process of detection, tracking, and cataloging space objects is known as Space Situational Awareness (SSA). Since the beginning of the space program, the U.S. Air Force Space Command has been charged with the tracking of various active satellites, de-activated satellites, and space debris for the purposes of cataloging and conjunction analysis. With decades of man-made objects being placed into various Earth orbits, the tracking of these objects becomes evermore difficult, with the probability for inaccurate object position estimates, improper object identification, and even object collisions increasing as each new object is introduced to the space environment. Current estimates of the number of objects needed to be tracked are around twenty thousand, while the U.S. resources availible to track them stand at about twenty globally positioned optical and phased array sensors known as the Space Surveillance Network (SSN).

For scenarios such as this in which there exists a disproportionate ratio between objects tracked and sensors available to observe them, a problem may arise in which a particular sensor may have multiple objects visible to it, but constraints prohibit it from observing all of those objects within the window where

they are visible. Additionally, problems may also arise in cases where large gaps exist between opportunities to make observations. This situation is often the case for satellite tracking problems, where ground-based sensors have limited fields of view and where the Earth's rotation and object dynamics can yield long periods before an observable pass will occur between an object and a particular sensor. These gaps in observations, mis-modeling of the orbital dynamics, sensor noise, and numerical errors may lead to potential divergence in the position estimation and uncertainty associated with that object. This could pose several tracking issues, including poor predictive capability, inability to associate measurement data with appropriate estimates (poor data association), and unrealistic or physically impossible estimate behavior.

In this situation, the need arises to make an intelligent decision of which object to observe and which to ignore, a process henceforth referred to as sensor tasking, or sensor network management. This decision making process could have significant consequences, particularly if it results in not observing objects which are infrequently availible to the sensors. Therefore, methods must be introduced which allow for sensors to make these difficult tasking decisions. These methods will generally revolve around calculating a utility metric which prioritizes objects for observation, and allocates sensor resources to best observe those priorities. In many cases, this priority will revolve around observing objects to maintain the most accurate state and uncertainty estimates for all objects being tracked, and typically rely on object state/uncertainty estimates, or other properties (object dynamics models, sensor models, etc.) in their calculation.

This implies that the methods of state/uncertainty estimation can play a vital role in the formation of such a utility metric. For the nonlinear dynamics present in space object state propagation, and the possible application of nonlinear measurements based on these state values, a nonlinear filter must be applied in order to obtain the state/uncertainty estimates necessary to not only monitor these objects, but also calculate potential tasking utility metrics. Nonlinear filtering and estimation techniques have been thoroughly researched in the field of satellite orbit determination, with particular applications to tracking resident space objects.

In the tracking process, these filters provide a probability density function (PDF) describing an object's state and uncertainty by calculating an estimated mean and covariance. Sensors (optical telescopes, radar, patched arrays, interferometers, and transponders) can provide measurements of a particular object's position at various times, which can be used within a nonlinear filter to update these estimates to more accurately reflect its location and uncertainty. However, different methodologies exist concerning both the updating of state and uncertainty estimates, as well as propagating them forward in the time span between updates. Previous work has shown in general that estimators which can better handle system nonlinearities result in better estimates than filters that rely on approximations or poor models of system nonlinearities.

Therefore, success in monitoring these objects relies not only on selecting an appropriate tasking method, but also on selecting the best possible nonlinear estimator. Furthermore, since tasking decisions will depend on metrics calculated from the state and covariance estimates provided by the particular estimator used, a coupling effect between tasking and estimation is produced. That is, a filter which provides superior estimates may in fact make better tasking decisions, resulting in fitter data which further improves its estimates over an inferior method. The purpose of this dissertation is to examine how this coupling can amplify performance differences between estimator types beyond those that are seen if the estimators are given identical data sets.

To study this coupling, various methods of estimation and sensor tasking are applied to a simplified simulation of the estimation and tasking components of the SSA tracking problem. In general terms, SSA encompasses a broad range of topics from detecting new space objects, characterizing and identifying existing objects, and understanding of the space environment, while incorporating methods in optical-based satellite detection, nonlinear estimation, sensor tasking, and data association. While this problem includes a broad and interconnected array of engineering topics, the scale of the problem is reduced by investigating the specific components of estimation and tasking, and assuming the data association, noise estimation, and additional components to be modeled as given quantities instead

of deterministically.

1.2 Review of Previous Literature

The process of monitoring resident space objects has been thoroughly researched, mainly concerning methods of orbit determination (estimation) and sensor resource allocation (sensor tasking). Details of current tasking methods for the SSA problem using the SSN are given in the work of Miller [3], while specific characteristics of the primary optical and Ground Based Radar (GBR) sensors comprising the SSN can be found in the work of Vallado and Griesbach. [4].

1.2.1 Sensor Tasking and SSA

Current tasking methods of this problem revolve around observation based on priority (such as atmospheric reentry, elapsed time since last observation, military interest, etc.) and do not incorporate motivations such as uncertainty/chaos reduction presented in these studies. These tasking methods separate the near-Earth orbit regimes of space objects into energy dissipation bins, and provide a suggested amount of observations (or tracks) per day. The motivation for this form of tasking came from a previous method of binning objects into Gabbard classes based on perigee and apogee height of the object's respective orbits [3]. This simple form of sensor allocation was later replaced by the current method based on a study conducted by Lockheed Martin [5] which found a more effective binning method than the previously used Gabbard classes. This new method divided objects in the satellite catalog into 11 energy dissipation rate bins (each with their own suggested tracks per day) based on the amount of atmospheric drag a particular satellite experiences.

While this current method of tasking has shown to work, recent publications have suggested using more advanced methods in sensor tasking in order to create more optimal tasking solutions to the SSA problem. Some of these methods

include tasking sensors to observe the most possible objects in some frame of time [6], or incorporating game theoretic approaches involving intelligent tracking responses based on the characteristics of objects being detected/tracked [7] and sensors tracking them [8]. These methods require some key assumptions about the types of objects being tracked (such as assuming they are all in low-Earth orbit [6]), and generally revolve around the awareness concept of the SSA problem, in which determining what specific objects are doing is more important than tracking as many objects as possible. For this reason, these studies will look to other methods of sensor tasking that are based more on overcoming the difficulties in observing many objects using few sensor resources. Due to this, the tasking methods selected for these studies should be driven by the motivation to maintain the lowest uncertainty (or highest information) among all objects tracked.

Fortunately, this type of approach has been thoroughly studied for general applications to sensor network management problems, which could be easily applied to the SSA problem. These tasking strategies gained tremendous popularity in the 1990's, especially with the work of Schmaedeke [9], which proposed a method of sensor tasking that would maximize information gained by each sensor. This measure of information was referred to as the relative information gain, which measured the difference in information of a Gaussian posterior about a Gaussian prior distribution utilizing covariance matrices. This relative information gain was in fact the same as the Shannon information, which reflects information gained about the state (in this case, the prior covariance) [10]. This form of information can be easily computed in conjunction with a Kalman filter algorithm (to gain the necessary Gaussian covariance estimates), whose application to sensor tasking problems has been illustrated in several recent works [11, 12, 13].

Additionally, others such as Tian et. al. [14] have suggested using similar information-theoretic approaches utilizing Fisher information. Unlike the relative measure of Shannon information, Fisher information, which represents the upper bound on information present in an unbiased estimator [15], is an absolute measure of information. Like Shannon information, this metric can be easily calculated given a prior and posterior covariance estimate provided by a sequential Kalman

filter algorithm [16, 17]. In both the cased of using Shannon information or Fisher information, the information-theoretic metric is calculated for each possible object-sensor pair, to which methods of linear programming are used to allocate which sensors will observe which objects [9, 14]. This application makes these methods myopic, or greedy in nature, due to the fact that they only account for information gained over one particular instance in time, and do not predict for consequences such as limited object access, or the tendency of uncertainty to diverge should observations occur infrequently.

However, should tasking decisions be based upon projecting the possible information gained over some finite *look ahead time*, a problem arises in computational costs associated with solving a potentially large scale dynamic programming problem. In fact, Hero et. al. [18] wrote an entire book focusing on these types of sensor network management problems, suggesting information metrics (such as Fisher information) as cost functions to drive dynamic programming solutions to the Markov Decision Process of sensor resource allocation. Methods such as receding horizon control [19] attempt to reduce the complexity of solving these problems by taking the fact that covariance estimates can be generated without the use of measurement data to create a partially observable Markov decision process that is less complicated to solve. Additionally, others have suggested using genetic algorithms to solve the large-scale optimization problems present in multi-object, multi-sensor allocation problems [20]. However, these studies have focused on problems that do not contain the computational complexity of the SSA problem, or require computational costs that would not be appropriate for the purposes of these studies.

Recently, Erwin et. al. [21] applied the use of Fisher information as a utility metric specifically to the SSA problem. In this approach, a myopic form of sensor tasking was implemented, which enabled tasking and allocation algorithms inside a small-scale SSA simulation at relatively low computational costs. This study showed the benefits of an application of Fisher information directly to the SSA problem, and how the sensor allocation process could be easily achieved with the conjunction of an extended Kalman filter to provide state and covariance estimates.

However, these studies (nor the ones previously mentioned) do not address how the quality of the estimates generated using the Kalman filter algorithm could effect the sensor tasking decisions, and furthermore overall performance of the sensor networks ability to effectively track all of the objects. Therefore, to better study solutions to the SSA problem, the method of estimation, and not only the tasking strategy must be investigated.

1.2.2 Estimation and Orbit Determination

Since the 1960's, many estimation methods have been created based on a recursive algorithm providing estimates of state and covariance in the presence of noise in both state dynamics and measurement data, known as the Kalman filter. Originating with a preliminary paper by Kalman [22], this algorithm used a multi-step method of filtering in which estimates were linearly propagated forward in time in a *forecast step*, and then refined within an *update step* upon receiving measurement data. These steps can be repeated an infinite number of times, allowing for state and uncertainty estimation to occur in real-time so long as measurement data continues to be collected.

Adaptations and advancements were eventually made to this preliminary sequential Kalman filter algorithm, including the ability to filter in the presence of nonlinear dynamics and measurement models known as the extended Kalman filter (EKF). This method proved beneficial for purposes of orbit determination (given the nonlinear dynamics of satellite motion) and was first implemented in the Apollo Moon landings [23]. Eventually, the EKF would be widely used within other nonlinear filtering problems, including modern aerospace applications in robotics and navigation, such as autonomous flight [24]. The EKF differed from the original linear Kalman filter by propagating uncertainty linearly using a state transition matrix, propagating the state estimate through the full nonlinear dynamics, and approximating nonlinear predicted measurement uncertainty using a first order Taylor series approximation about the current state estimate [25]. This approach is effective so long as the the propagation time in the forecast step is short enough

so that second order or higher errors from the linearization are small.

Nearly forty years after the inception of the EKF, a new adaptation of the Kalman filter algorithm known as the unscented Kalman filter (UKF) claimed to better approximate system nonlinearities, and therefore yield better estimates than the EKF for nonlinear systems [26]. The UKF does this by using a small distribution of *sigma points* that are propagated directly through the nonlinear equations of motion, which provide state and uncertainty estimates in the forecast step, and measurement approximations in the update step using statistical analysis. The result is that the state and uncertainty estimates calculated using these methods do not necessitate linear approximations of system nonlinearities, and could therefore achieve second order or higher accuracy [27].

Typically, when presented with a consistent time interval of measurements, the UKF has produced better estimates than the EKF for nonlinear systems ranging from simple pendulums [28] to more complex problems in orbit determination [28, 29] and autonomous flight [30]. Depending on the extent of the nonlinearity in the system dynamics or measurement models, linearization errors resulting from the EKF could be amplified, leading in to a quicker divergence in state estimate error when using an EKF opposed to a UKF. These errors may play an essential role in the performance discrepancies seen when comparing the two filters [30]. Moreover, it has also been shown that, for the case of orbit determination, the EKF has trouble producing accurate estimates as time between measurements increases [28], reinforcing the conjecture that as propagation time in the forecast step increases, linearization errors grow and result in poor estimation for the EKF.

In an attempt to try to better propagate and model uncertainty in nonlinear filtering problems, an additional class of nonlinear filters were derived which utilized a Gaussian mixture model (GMM) approach to the Bayesian estimation problem [31]. In this approach, an initial Gaussian probability density function (PDF) described by a mean and covariance can be split into multiple Gaussian PDFs to be propagated and updated in parallel. The goal of the GMM is to better describe the actual non-Gaussian PDF resulting from propagation of the initial PDF through

a nonlinear transformation than could be possible using a single Gaussian PDF (such as the case using an EKF, and possibly a UKF). The GMM filter was eventually adapted to be applied within an EKF [32] and later a UKF, and has been tested within orbit determination problems to show better results of uncertainty estimates when compared to a standard UKF [33]. The GMM approach to filtering has also been advanced further by Terejanu et. al. [34] who presented a method of updating weights to GMM components, further improving the non-Gaussian PDF modeling made possible using a GMM.

One of the most recent additions to the GMM approach has come in the application of an adaptive entropy-based Gaussian-mixture information synthesis (AEGIS) filter, which has showed to outperform the UKF in the propagation of uncertainty in orbit determination problems [35, 36]. The AEGIS filter adapts the GMM to incorporate a splitting algorithm within the UKF propagation of uncertainty, so that when a degree of nonlinearity is detected a single Gaussian PDF can be split into a GMM of several Gaussian PDFs. This GMM more accurately reflects the actual PDF in the presence of these nonlinearities, implying state and uncertainty estimates are more accurate as well.

The central hypothesis of the work presented here is that the improved estimates provided by a UKF over an EKF, and furthermore an AEGIS filter over a UKF will not only help the orbit determination component of a satellite tracking problem, but also provide better tasking decisions and sensor schedules given these improvements.

1.3 Research Contributions

The focus of this work is to examine the effect that nonlinear estimation has on sensor tasking, tested specifically within the estimation and tasking components of a satellite tracking problem. This research models the estimation and tasking components of SSA, reflecting general tasking/estimation coupling trends observable in a wide variety of nonlinear tracking problems. In general terms, SSA encompasses

a broad range of topics from detecting new space objects, characterizing and identifying existing objects, and understanding of the space environment, while also incorporating methods in optical-based satellite detection, nonlinear estimation, sensor tasking, and data association. While this problem includes a broad and interconnected array of engineering topics, the scale of the problem can be reduced by investigating the specific relationship between estimation and tasking, and assuming the data association (i.e. understanding which object is being observed), noise estimation, and additional components are known a priori.

While many publications have compared methods of nonlinear estimation for orbit determination problems, and others have suggested methods for sensor network management, none have studied the coupling of the two. That is, a filter which provides superior estimates may in fact make better tasking decisions, resulting in superior data which further improves its estimates over an inferior method. Therefore, the primary purpose (and largest contribution of original thought) of this dissertation is to examine how this coupling can amplify performance differences between estimator types beyond those that are seen if the estimators are given identical data sets.

In order to accomplish this, this work also makes several additional contributions to methods of dynamic sensor tasking as well as nonlinear estimation. For nonlinear estimation, while much previous research has compared performance of the EKF and UKF applied to nonlinear tracking problems, very little exists in examining the AEGIS filter. Recent work has shown that the AEGIS filter outperforms a UKF (and is therefore assumed it would outperform an EKF) [35, 36], but no work has been done that integrated the AEGIS filter into a sensor allocation algorithm and compares its performance to an EKF and UKF under those conditions. Since the satellite tracking simulation in this work will include large time spans of propagating estimates in the abscense of measurement data, it will further investigate the improvement in uncertainty propagation and updating gained using the AEGIS filter as opposed to an EKF or UKF.

In regards to sensor tasking, two myopic information-based metrics utilizing

Fisher information gain (FIG) [14] and Shannon information gain (SIG) [11] will be investigated. These particular information measures were chosen due the legacy of their applications in sensor network management strategies and ease of extraction given state/uncertainty estimates provided by nonlinear filters [12, 21]. Additionally, they represent both an absolute and relative measure of information gain. While the application and effectiveness of these two methods have been studied individually, no current work investigates how these two information-based sensor tasking strategies compare to each other. This work will determine whether there are advantages to using absolute or relative measures of information, as well as illustrates how they may be extracted from a UKF or AEGIS filter, and not just the EKF as previous research showed.

To accompany the two information-theoretic tasking approaches, a third new type of tasking method is studied. This metric is based on maintaining stability of the error dynamics from the propagating and updating of estimates conducted within the application of the nonlinear filters. The metric in question will be calculated using elements of largest Lyapunov exponent (LLE) estimation [37], a tool commonly used to assess state-space stability of dynamic systems and/or experimental data, and has not been previously suggested as a possible utility metric for sensor tasking problems. The motivation to consider this metric is based on work conducted by Rauf et al. [38] which describes how to calculate these stability measures for the error dynamics in filters using the root mean square error (which can be extracted from a covariance estimate provided by a filter). It is hypothesized that this metric would create tasking decisions prioritizing objects which showed the greatest trends of divergence in their estimated uncertainty. This dissertation will therefore be the first work which shows how to extract this stability-based metric within an EKF, UKF, or AEGIS filter algorithm, as well as how to use it as a tasking utility metric. Furthermore, this will be the first work to compare this new method to the previously studied tasking strategies utilizing FIG and SIG.

1.4 Dissertation Outline

The doctoral research presented here is organized in the following manner. In Chapter 2, the two-body satellite dynamics and nonlinear range-angle measurement data used to create a space object tracking model are detailed. This section serves to illustrate how space objects and sensors will be populated and propagated within a simplified planar two-body acceleration force model (for orbiting objects) using a rotating Earth model (for ground-based sensors). Additionally, it describes how a sensor's field of regard is defined, and how it is used to determine sets of objects available for observation and sensors available to observe them at each simulation time step.

Chapter 3 begins by providing a brief background on probability theory (concentrating on Gaussian distributions) and signal filtering, before detailing the linear Kalman filter. Two nonlinear extensions of the linear Kalman filter are then detailed (the EKF and UKF), highlighting the application of these filters to a multi-object, multi-sensor filtering problem, as well as how they differ in methods of approximating nonlinear covariance propagation and measurement functions. This is followed by the introduction to GMMs, and how they are applied to nonlinear filtering by introducing the AEGIS filter. In this section, general GMM properties, methods of splitting GMM components, detection of system nonlinearity, merging GMM components, and their application within a UKF bases filtering scheme will be discussed. Finally, the chapter ends with detailed recursive algorithms for the EKF, UKF, and AEGIS filters with respect to their applications within a nonlinear multi-object tracking problem.

In Chapter 4, the concept of sensor tasking is illustrated, including times when tasking decisions are made, as well as constraints imposed on the tasking process. Three covariance-based tasking methods will then be discussed, starting with two information-theoretic approaches and concluding with a new stability-based approach. Fisher information gain is first discussed, which reflects an absolute gain in information, followed by Shannon information, a relative measure of information gained about the state of a system. A new method is then illustrated that deter-

mines the error-stability for a particular object in the form of estimating a largest Lyapunov exponent. In each case, details for calculating these metrics within an EKF, UKF, and AEGIS filter are presented.

Chapter 5 discusses how performance of the estimation/tasking combinations is evaluated after completion of a simulation, with simulation set-up, results and analysis presented in Chapter 6. Results are based on identical simulations for two test cases. The simulations incorporate each of the three tasking methods applied to each of the three nonlinear filters, with an additional ideal simulation conducted for each of the three filters using no tasking (i.e. all sensors collect data for all objects within their field of regard at all simulation time steps). These simulations are conducted for two test cases of low initial error (ideal for most practical satellite tracking applications) and a high initial error case (highlighting performance in non-ideal conditions).

Finally, in Chapter 7 conclusions are drawn as to the coupling effects of the particular methods of tasking and estimation, in which the average estimation error over all satellites will be held as the paramount indication of which combination of methods was the most ideally suited towards this particular facet of the satellite tracking problem. In addition, performance results with respect to each method of tasking and estimation will be discussed, highlighting which of these methods would be most ideal to a real-world satellite tracking problem. This chapter concludes with recommendations for future work that can be derived from these studies, with applications both to the problem of satellite tracking and general nonlinear tracking problems.

Chapter 2

Dynamics and Measurement Models

To construct a simulation of the estimation and tasking components of the SSA problem, the equations of motion of both space objects tracked and sensors used to track them, as well as models of sensors measurement data are defined. These models reflect realistic (though simplified) trajectories of orbiting space objects as well as ground and space based sensor positions and fields of view throughout the duration of the simulation. All states of objects and sensors will be based on an inertial $x - y$ coordinate system with the Earth's center at its origin (i.e. this coordinate frame it will stay fixed with respect to a constant rotating Earth model). This system was chosen over a more realistic three dimensional system for computational simplicity. Since the motivation for these studies is to determine the coupling between estimation and sensor tasking within the nonlinear framework of orbital mechanics, the use of a third dimension only adds increased computational complexity, and is therefore not necessary to complete the objectives of this research. The two-dimensional Earth is also assumed to be described by a circle, containing no oblateness such that the force of gravity is considered constant at a certain displacement from the origin. A generic illustration of this system populated with space objects and various sensors which would be tasked to track them (both space and ground-based) similar to this SSA simulation can be seen in Figure 2.1.

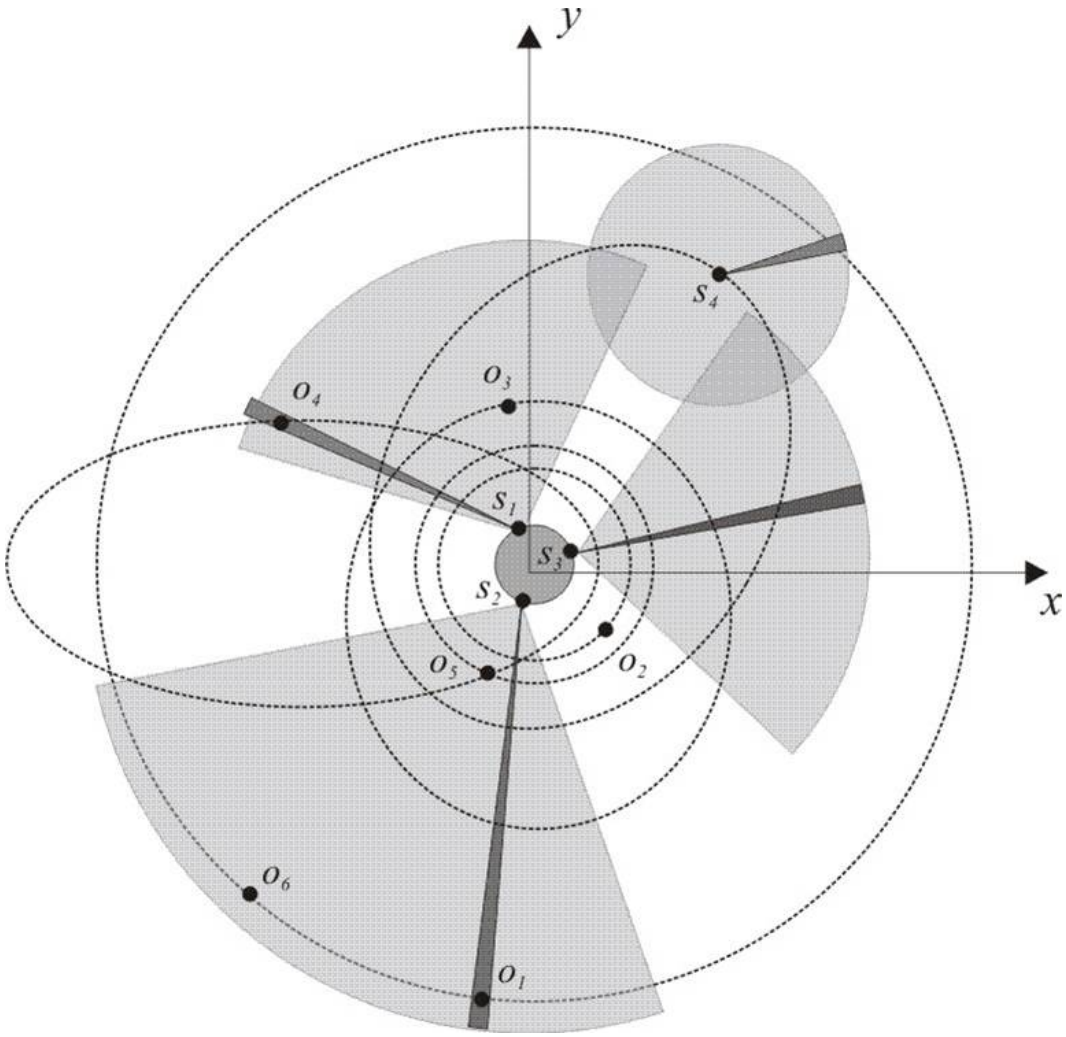

Figure 2.1. Illustration of a simple SSA system necessitating sensor tasking for 6 orbiting objects O_i $(i = 1, \ldots, 6)$ to be tracked, and sensors S_j $(j = 1, \ldots, 4)$ to track them (3 ground, 1 orbiting).

2.1 Object Dynamics

To model object motion and sensor locations, a series of N_s 2-D objects in the set $\mathcal{O} = \{o_1, \ldots, o_{N_s}\}$ are modeled, such that their distribution is bimodal with the majority of orbits being within a proximity of low-Earth orbit (LEO) or geostationary Earth orbit (GEO). Additionally, object eccentricity is determined from a one degree of freedom chi-square distribution which places most of the objects in near circular orbits. For these studies, the primary assumption in the dynamics of each object tracked is that measurements of their location, as well as state/uncertainty

updates (resulting from sensor tasking) are only possible at discrete time intervals. Therefore, for a particular object, it is assumed that measurements are only possible at times $t_k = k\Delta t,\ k = 1, 2, \ldots$ where Δt is a constant time step. At these discrete times, the motion of an object is also described in state-variable form, to which the n element[1] state vector for the given object i at time k is defined as

$$\vec{x}_{i,k} = \begin{bmatrix} x_{i,k} \\ y_{i,k} \\ \dot{x}_{i,k} \\ \dot{y}_{i,k} \end{bmatrix} \tag{2.1}$$

Assuming that each space object (and orbiting sensor) meets the criteria for a two-body force model [39]

- The mass of the space object is negligible compared to that of the Earth
- The coordinate system for the simulation is inertial
- The bodies of the satellite and attracting body are spherical, with uniform density
- No additional external forces act on the system except gravity, which acts along the line joining the centers of the space object and the Earth

their corresponding equations of motion are described by

$$\dot{\vec{x}}_{i,k} = \begin{bmatrix} \dot{x}_{i,k} \\ \dot{y}_{i,k} \\ \ddot{x}_{i,k} \\ \ddot{y}_{i,k} \end{bmatrix} = f\left(\vec{x}_{i,k}\right) + \vec{w}_{i,k} = \begin{bmatrix} \dot{x}_{i,k} \\ \dot{y}_{i,k} \\ -\mu_E x_{i,k}/r_{i,k}^3 \\ -\mu_E y_{i,k}/r_{i,k}^3 \end{bmatrix} + \vec{w}_{i,k} \tag{2.2}$$

where $r_{i,k} = \sqrt{x_{i,k}^2 + y_{i,k}^2}$, μ_E is the constant gravitational parameter of Earth, the subscript i refers to an index in the set of N_s objects being tracked, and $\vec{w}_{i,k}$ is the object's process noise vector with covariance $Q_{i,k} = \vec{w}_{i,k}\vec{w}_{i,k}^T$. For these studies, process noise is taken to be any unmodeled disturbance to an object's known

[1] In this case, the size of the state is $n = 4$

equations of motion, which can take the form of unmodeled perturbations or errors which exist in numerical propagations (such as applications of a linear state transition matrix, or numerical integration).

Equation 2.2 can be propagated forward from time step k to $k+1$ using one of two methods. First, it can be propagated by direct numerical integration of Equation 2.2, a transformation represented by the function $\mathcal{F}(\cdot) = \int_{t_k}^{t_{k+1}} f\left(\vec{x}, t\right) dt$

$$\vec{x}_{i,k+1} = \mathcal{F}\left(\vec{x}_{i,k}\right) \tag{2.3}$$

or it can be propagated as a deterministic system using a state transition matrix $\Phi_{i,k|k+1}$

$$\vec{x}_{i,k+1} = \Phi_{i,k|k+1}\vec{x}_{i,k} \tag{2.4}$$

where $\Phi_{i,k|k+1}$ is calculated from numerical integration of the equation

$$\dot{\Phi}_{i,t_k|t} = \left(\frac{\partial f}{\partial \vec{x}}\right)_{i,t} \Phi_{i,t_k|t} \tag{2.5}$$

from time-step $k \rightarrow k+1$. In this propagation, $\Phi_{i,k|k+1}$ represents the final value of $\Phi_{i,t_k|t}$ from the propagation of Eq. (2.5), with the initial condition $\Phi_{i,t_k|t_k} = I_{n\times n}$. Additionally, the propagation time t is some time within the time steps k and $k+1$. For this particular $n = 4$ state problem, the $n \times n$ matrix of partials in Eq. 2.5 is defined as

$$\left(\frac{\partial f}{\partial \vec{x}}\right)_{i,t} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \mu_E * \left(3x_{i,t}^2/r_{i,t}^5 - 1/r_{i,t}^3\right) & 3x_{i,t}y_{i,t}\mu_E/r_{i,t}^5 & 0 & 0 \\ 3x_{i,t}y_{i,t}\mu_E/r_{i,t}^5 & \mu_E * \left(3y_{i,t}^2/r_{i,t}^5 - 1/r_{i,t}^3\right) & 0 & 0 \end{bmatrix} \tag{2.6}$$

2.2 Sensor Dynamics

A series of M_s sensors given by the set $\mathcal{S} = \{s_1, \ldots, s_{M_s}\}$ are modeled, such that they may either represent a ground sensor, or an orbiting sensor. The location of

the sensors in the simulation is assumed to be known with no uncertainty, and can be described by

$$\vec{s}_{j,k} = \begin{bmatrix} x^s_{j,k} \\ y^s_{j,k} \\ \dot{x}^s_{j,k} \\ \dot{y}^s_{j,k} \end{bmatrix} \tag{2.7}$$

with equations of motion

$$\dot{\vec{s}}_{j,k} = g_o\left(\vec{s}_{j,k}\right) = \begin{bmatrix} \dot{x}^s_{j,k} \\ \dot{y}^s_{j,k} \\ -\mu_E x^s_{j,k} / \left(r^s_{j,k}\right)^3 \\ -\mu_E y^s_{j,k} \left(r^s_{j,k}\right)^3 \end{bmatrix} \tag{2.8}$$

if the index j corresponds to an orbiting satellite, where the sensor position $r^s_{j,k} = \sqrt{\left(x^s_{j,k}\right)^2 + \left(y^s_{j,k}\right)^2}$. Furthermore, should the index j correspond to a ground sensor the equations of motion are given by

$$\dot{\vec{s}}_{j,k} = g_g\left(\vec{s}_{j,k}\right) = \begin{bmatrix} \dot{x}^s_{j,k} \\ \dot{y}^s_{j,k} \\ -\omega_E^2 x^s_{j,k} \\ -\omega_E^2 y^s_{j,k} \end{bmatrix} \tag{2.9}$$

where ω_E is the constant rotational rate of the Earth. To propagate these sensor equations of motion forward in time, the same techniques can be applied as with the case of object equations of motion. In this case, should the index j reflect an orbiting sensor, this would yield propagation using

$$\vec{s}_{j,k+1} = \mathcal{G}_j\left(\vec{s}_{j,k}\right) \tag{2.10}$$

where $\mathcal{G} = \int_{t_k}^{t_{k+1}} g_o\left(\vec{s}, t\right) dt$, or

$$\vec{s}_{j,k+1} = \Phi_{j,k|k+1} \vec{x}_{j,k} \tag{2.11}$$

where

$$\dot{\Phi}_{j,t_k|t} = \left(\frac{\partial f}{\partial \vec{s}}\right)_{j,t} \Phi_{j,t_k|t} \tag{2.12}$$

and

$$\left(\frac{\partial f}{\partial \vec{s}}\right)_{j,t} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \mu_E * \left\{ \frac{3\left(x^s_{j,t}\right)^2}{\left(r^s_{j,t}\right)^5} - \frac{1}{\left(r^s_{j,t}\right)^3} \right\} & \frac{3x^s_{j,t}y^s_{j,t}\mu_E}{\left(r^s_{j,t}\right)^5} & 0 & 0 \\ \frac{3x^s_{j,t}y^s_{j,t}\mu_E}{\left(r^s_{j,t}\right)^5} & \mu_E * \left\{ \frac{3\left(y^s_{j,t}\right)^2}{\left(r^s_{j,t}\right)^5} - \frac{1}{\left(r^s_{j,t}\right)^3} \right\} & 0 & 0 \end{bmatrix} \tag{2.13}$$

However, if the index j represents a ground-based sensor, there exists a closed form solution to propagating Eq. 2.9 since it is a linear differential equation with solution [40]

$$\vec{s}_{j,k+1} = \begin{bmatrix} R_E \cos\left(\omega_E \left\{t_k + \Delta t\right\} + \theta_{j,0}\right) \\ R_E \sin\left(\omega_E \left\{t_k + \Delta t\right\} + \theta_{j,0}\right) \\ -R_E\omega_E \sin\left(\omega_E \left\{t_k + \Delta t\right\} + \theta_{j,0}\right) \\ R_E\omega_E \cos\left(\omega_E \left\{t_k + \Delta t\right\} + \theta_{j,0}\right) \end{bmatrix} \tag{2.14}$$

where R_E is the constant radius of the Earth, and $\theta_{j,0}$ is the sensors initial angle relative to the positive x-axis such that $\theta_{j,0} = \cos^{-1}\left(x^s_{j,0}/R_E\right)$ with quadrant checks applied if $x^s_{j,0} < 0$.

2.3 Sensor Model

Observations will be made by a total of M_s sensors, containing ground based and orbiting space based sensors. A simple illustration of this system can be seen in Figure 2.1, while the sensors field of regard Γ_j (bounded by a maximum range Δ_j and half-angle with respect to the local vertical[2] Ψ_j) can be seen in Figure 2.2. The sensors field of regard reflects an area in the Earth-centered inertial Cartesian $x-y$ coordinate system in which a sensor can gather position data on an object.

[2]In the case of space-based sensors, the local vertical is assumed to be parallel to the orbiting sensors position vector measured from the center of the Earth inertial $x-y$ coordinate system to the sensor.

This position will be reflected as one range measurement (in kilometers), and one elevation measurement (in radians).

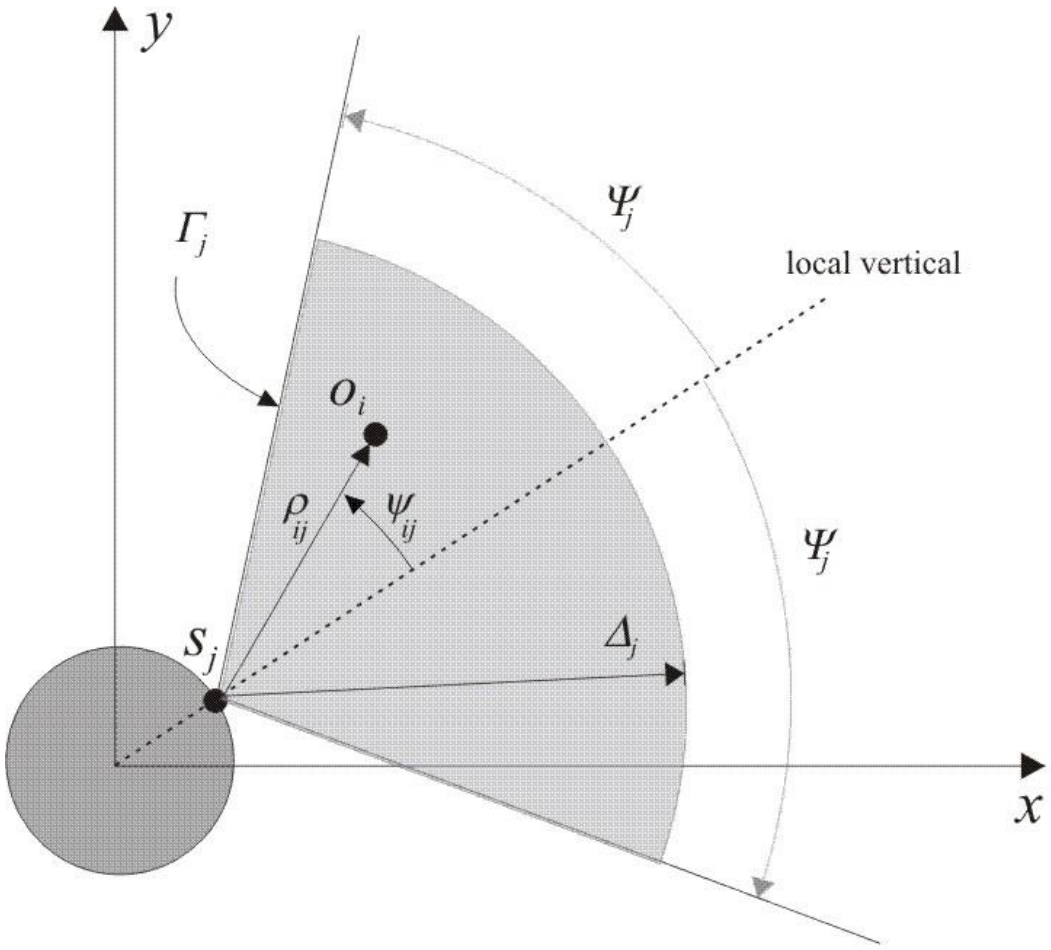

Figure 2.2. Illustration of SSA system sensor measurements from sensor S_j of object O_i with *field of regard* Γ_j spanning a range of Δ_j and half-angle Ψ_j (represented by shaded region). In this figure, the $x-y$ axis reflects an Earth-centered inertial system.

Measurements $\vec{y}_{(i,j),k}$ of a satellite i by sensor j are deemed as available measurements if the object i is positioned within the field of regard, $\Gamma_{j,k}$ of sensor j at time k, as illustrated in Figure 2.2. The measurement taken by a sensor on an object at a given time is represented by a nonlinear measurement function

$$\vec{y}_{(i,j),k} = \begin{bmatrix} h_\rho\left(\vec{x}_{i,k}, \vec{s}_{j,k}\right) \\ h_\psi\left(\vec{x}_{i,k}, \vec{s}_{j,k}\right) \end{bmatrix} + \vec{v}_{j,k} = \begin{bmatrix} \sqrt{\left(x_{i,k} - x^s_{j,k}\right)^2 + \left(y_{i,k} - y^s_{j,k}\right)^2} \\ \tan^{-1}\frac{y_{i,k}-y^s_{j,k}}{x_{i,k}-x^s_{j,k}} - \tan^{-1}\frac{y^s_{j,k}}{x^s_{j,k}} \end{bmatrix} + \begin{bmatrix} v^\rho_{j,k} \\ v^\psi_{j,k} \end{bmatrix} \tag{2.15}$$

where $h_\rho\left(\vec{x}_{i,k}, \vec{s}_{j,k}\right) = \rho_{(i,j),k}$, and $h_\psi\left(\vec{x}_{i,k}, \vec{s}_{j,k}\right) = \psi_{(i,j),k}$. Additionally, $\vec{v}_{j,k}$ is a particular sensors measurement noise vector with the measurement noise covariance

$$R_{j,k} = \begin{bmatrix} \left(v^\rho_{j,k}\right)^2 & 0 \\ 0 & \left(v^\psi_{j,k}\right)^2 \end{bmatrix} \tag{2.16}$$

Measurement noise, similar to process noise, is viewed as any disturbance related to the actual measurement of one or more of an object's characteristics (such as range), which reflects a source of empirical error in the tracking problem. For these studies both the process and measurement noise are assumed to be purely additive zero-mean Gaussian noise. Additionally, quadrant checks will be necessary so solve Eq. (2.15), should a sensors maximum half-angle $\Psi_j > 90^\circ$[3].

A measurement of space object i by sensor j is *possible* if the space object is positioned within the field of regard of the sensor at some time step, as illustrated in Figure 2.2. At each time step, a set of indices representing available objects

$$\mathcal{O}^a_k = \left\{ i : \ \rho_{(i,j),k} < \Delta_j \text{ and } \left|\psi_{(i,j),k}\right| < \Psi_j, \ \forall j \right\} \tag{2.17}$$

and sensors available to observe them

$$\mathcal{S}^a_{i,k} = \left\{ j : \ \rho_{(i,j),k} < \Delta_j \text{ and } \left|\psi_{(i,j),k}\right| < \Psi_j \right\} \tag{2.18}$$

detailing which sensors have the possibility to observe each object at a given measurement time. The tasking algorithms used will select the best combination of instantaneously allowed observations given by these sets, as described in Chapter 4. This implies that even though an object is in the set $\mathcal{O}^a_k$, it does not guarantee that it will be tasked for observation.

[3]This quadrant check can be achieved using an atan2 calculation method.

It is important to note that when determining these available object-sensor pairs, knowledge of which object is being observed is assumed to be known at all times. This is done to maintain the focus of these works strictly on the relationship between nonlinear estimation and sensor tasking, while excluding all other variables that could alter performance due to the selection of a filter/tasking method. This implies that for these particular studies, data association techniques would introduce an additional variable affecting the performance of the estimation-tasking pairs (should they be filter-driven, such as the methods suggested in the work of Kalandros et. al.[11]), and therefore are not included in the closed loop estimation and tasking algorithm. However, it should be noted that the concept of data association is critical in real-world applications of satellite tracking, such as SSA.

Chapter 3

Nonlinear Estimation

3.1 The Gaussian Distribution

Three candidate nonlinear estimators, an EKF, UKF, and AEGIS filter are implemented in these studies. The three estimators use similar Kalman filter-based algorithms, but contain many differences in the methods of propagating and updating their estimates. In each estimator, from time step iterations k to $k+1$, initial optimal state $\left(\hat{X}^*_{i,k}\right)$ and covariance $\left(\hat{P}^*_{i,k}\right)$ estimates will be propagated forward in time to yield forecast state $\left(\hat{X}^f_{i,k+1}\right)$ and covariance $\left(\hat{P}^f_{i,k+1}\right)$ estimates. Should measurement data be available for a particular object, it will then be subject to an update of these forecast estimates to yield a new optimal state $\left(\hat{X}^*_{i,k+1}\right)$ and covariance $\left(\hat{P}^*_{i,k+1}\right)$ estimate.

In each of these steps in the three filters implementation, the state (also referred to as the mean) and covariance estimates are used to describe a Gaussian PDF of object i at time k in the form

$$p^g\left(\vec{x}_{i,k}; \hat{X}^*_{i,k}, \hat{P}^*_{i,k}\right) = \left|2\pi\hat{P}^*_{i,k}\right|^{-1/2} \exp\left\{-\frac{1}{2}\left(\vec{x}_{i,k} - \hat{X}^*_{i,k}\right)^T \left(\hat{P}^*_{i,k}\right)^{-1} \left(\vec{x}_{i,k} - \hat{X}^*_{i,k}\right)\right\} \tag{3.1}$$

where the covariance $\hat{P}^*_{i,k}$ is symmetric, positive definite, with diagonal elements representing the variance $\left(\left\{\hat{\sigma}^n_{i,k}\right\}^2\right)$ of the n^{th} state variable. In general the covari-

ance matrix can also be described by the standard deviations $\left(\hat{\sigma}_{i,k}^{n}\right)$ with respect to each state variable in the form (specifically for the state variable described by Eq. 2.1)

$$\hat{P}_{i,k}^{*} = \begin{bmatrix} \left(\hat{\sigma}_{i,k}^{x}\right)^2 & \hat{\sigma}_{i,k}^{y}\hat{\sigma}_{i,k}^{x} & \hat{\sigma}_{i,k}^{\dot{x}}\hat{\sigma}_{i,k}^{x} & \hat{\sigma}_{i,k}^{\dot{y}}\hat{\sigma}_{i,k}^{x} \\ \hat{\sigma}_{i,k}^{x}\hat{\sigma}_{i,k}^{y} & \left(\hat{\sigma}_{i,k}^{y}\right)^2 & \hat{\sigma}_{i,k}^{\dot{x}}\hat{\sigma}_{i,k}^{y} & \hat{\sigma}_{i,k}^{\dot{y}}\hat{\sigma}_{i,k}^{y} \\ \hat{\sigma}_{i,k}^{x}\hat{\sigma}_{i,k}^{\dot{x}} & \hat{\sigma}_{i,k}^{y}\hat{\sigma}_{i,k}^{\dot{x}} & \left(\hat{\sigma}_{i,k}^{\dot{x}}\right)^2 & \hat{\sigma}_{i,k}^{\dot{y}}\hat{\sigma}_{i,k}^{\dot{x}} \\ \hat{\sigma}_{i,k}^{x}\hat{\sigma}_{i,k}^{\dot{y}} & \hat{\sigma}_{i,k}^{y}\hat{\sigma}_{i,k}^{\dot{y}} & \hat{\sigma}_{i,k}^{\dot{x}}\hat{\sigma}_{i,k}^{\dot{y}} & \left(\hat{\sigma}_{i,k}^{\dot{y}}\right)^2 \end{bmatrix} \tag{3.2}$$

Using this, the Gaussian uncertainty distribution of each variable is defined by

$$\begin{matrix} p^{g}\left(x_{i,k}; \hat{x}_{i,k}^{*}, \left\{\hat{\sigma}_{i,k}^{x}\right\}^2\right) \\ p^{g}\left(y_{i,k}; \hat{y}_{i,k}^{*}, \left\{\hat{\sigma}_{i,k}^{y}\right\}^2\right) \\ p^{g}\left(\dot{x}_{i,k}; \hat{\dot{x}}_{i,k}^{*}, \left\{\hat{\sigma}_{i,k}^{\dot{x}}\right\}^2\right) \\ p^{g}\left(\dot{y}_{i,k}; \hat{\dot{y}}_{i,k}^{*}, \left\{\hat{\sigma}_{i,k}^{\dot{y}}\right\}^2\right) \end{matrix} \tag{3.3}$$

The Gaussian PDF, while meeting the general requirements of all PDFs[1] has many unique properties which can make it an advantage or a hindrance to describe a PDF of a randomly distributed variable. First, a Gaussian distribution (also called a normal distribution) follows very closely to many naturally occurring distributions, mainly due to the fact that an overall collection of small independent stochastic variables of equal magnitude approaches a Gaussian distribution as the number of variables increases [42]. Second, Gaussian distributions are mathematically convenient due to the fact that when a linear transformation is applied to a Gaussian distribution, the resulting distribution remains Gaussian [43]. This convenience is exploited in the implementation of the linear Kalman filter algorithm (described in Section 3.2.2) due for the necessity for means and covariances estimated in the Kalman filter to remain Gaussian. However, this may pose problems when filtering must be achieved within nonlinear systems, since an initially Gaussian distribution propagated through a nonlinear transformation will not remain Gaussian. In this scenario, a Gaussian mean and covariance may provide a poor fit to describe the new non-Gaussian PDF.

[1]The PDF is always positive and has a value of one when integrated over its support set [41].

3.1.1 Gaussian PDFs Subject to Nonlinear Transformations

To illustrate the effects nonlinear transformations have on Gaussian PDFs, the specific transformation represented by the integration of an initially Gaussian PDF through satellite orbit dynamics (described in Eq. 2.2) is investigated. In this example, a normal distribution of particles representing an uncertainty distribution of a space object is created, such that the space object is located at the periapsis of the orbit, also functioning as the mean of the distribution given by

$$[x_m, y_m, \dot{x}_m, \dot{y}_m] = [r_p, 0, 0, v_p] \tag{3.4}$$

where r_p is the orbit periapsis radius $r_p = a(1-e) = 10{,}000$ km. For this particular example, the eccentricity of the orbit $e = 0.2$ and the velocity at periapsis can be calculated from $v_p = \sqrt{2\mu_E\left(1/r_p - 1/2a\right)}$. Additionally, the standard deviations for each state variable are chosen as

$$\left[\sigma^x, \sigma^y, \sigma^{\dot{x}}, \sigma^{\dot{y}}\right] = [10km, 10km, 0.05km/s, 0.05km/s] \tag{3.5}$$

Therefore, using Eqs. 3.3 - 3.5, an initial normal distribution of space objects is created, which is shown in Figure 3.1. In this figure, the 3σ error ellipse about the mean illustrates that almost all of the particles comprising this Gaussian distribution are located within this error ellipse. This is to be expected since approximately 99.7% of points within a Gaussian distribution should lie within its 3σ bounds[2].

Once this Gaussian distribution of particles is propagated through the nonlinear orbit dynamics in Eq. 2.2, the resulting distribution is shown in Figure 3.2. After propagation through the nonlinear system, a Gaussian PDF no longer provides as good an approximation to the uncertainty distribution of the particles as it did for the initial distribution[3]. Additionally, while the initial distribution was roughly

[2] In this case, these bounds are the error ellipse created by semi major and semi minor axes σ^x and σ^y

[3] A similar effect could be seen using other orbital coordinate systems (e.g. Keplerian, spher-

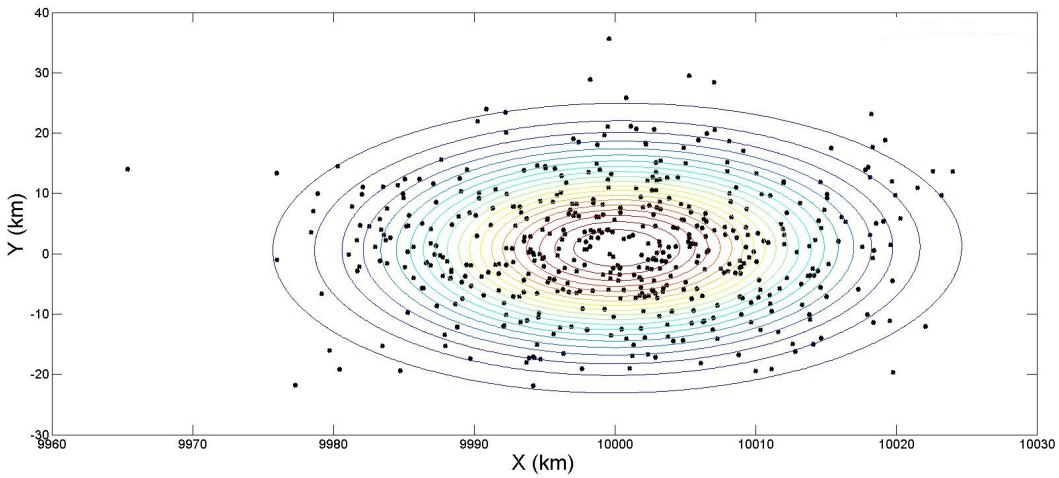

Figure 3.1. Illustration of initial Gaussian uncertainty distribution of space object and contour overlay of PDF evaluation using Eq. 3.1

elliptical (characteristic of a two-dimensional Gaussian PDF), the propagated distribution is now warped to be more banana shaped, a phenomenon commonly seen in orbit uncertainty propagation [35]).

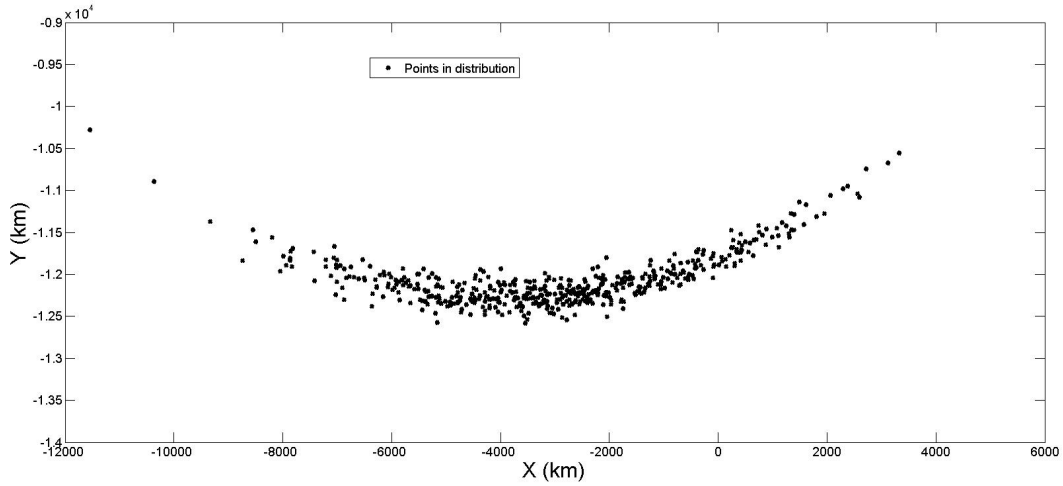

Figure 3.2. Illustration of initial Gaussian uncertainty distribution propagated through nonlinear orbit dynamics

In this case, a Gaussian PDF would not accurately model the actual PDF of the distribution, as can be seen in Figures 3.3 and 3.4. In these figures, the actual location of the data points (placed in 25 equally spaced bins spanning the

ical, etc.) since each have nonlinear equations of motion.

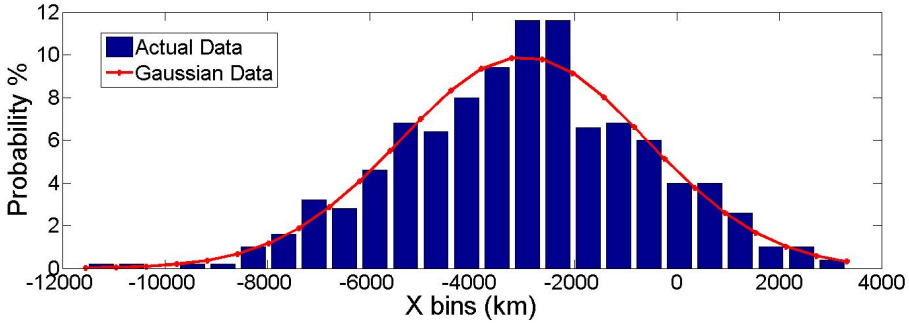

Figure 3.3. x-variable PDF of propagated distribution of space objects (binned into 25 equally spaced x-position locations) with corresponding Gaussian PDF overlay.

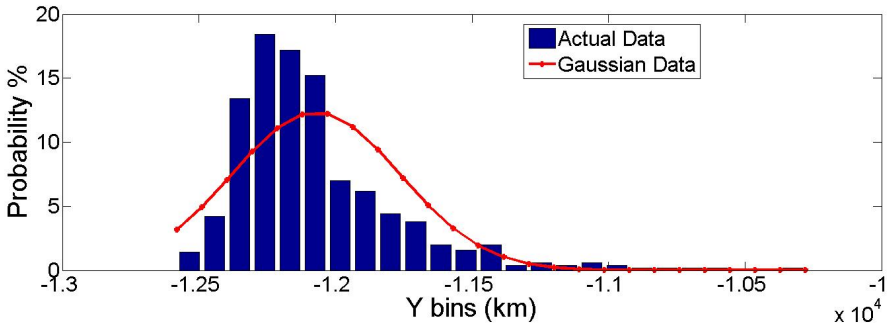

Figure 3.4. y-variable PDF of propagated distribution of space objects (binned into 25 equally spaced y-position locations) with corresponding Gaussian PDF overlay.

total dispersion in x and y locations of the data points) differs from the prediction of a Gaussian distribution with the same x and y mean and standard deviation. This observation is important due to the fact that all nonlinear filters used in these studies require Gaussian distributions to describe space object uncertainty models which will obviously not be Gaussian. However, the way in which they propagate and define these Gaussian distributions differs, leading some estimators to provide approximate Gaussian uncertainty models which better suit the actual non-Gaussian uncertainty of the objects as opposed to others. This concept will be revisited at later sections in the chapter when discussing the UKF and AEGIS filter, and how they account for these issues.

3.2 Wiener Filtering and the Linear Kalman Filter

Originating from statistical problems concerning communications and control, the process of "filtering" data, in which a random signal is separated into relevant data and disturbances (i.e. noise), made great advancements in the 1960s, mainly with the introduction of the Linear Kalman Filter [22]. The Kalman Filter, unlike its predecessors, was able to create an easily applicable and robust algorithm for filtering noisy signals out of linear systems.

3.2.1 The Wiener Filter

The roots and motivation for the Kalman Filter stem from previous work done by Wiener [44] on the subject of random signal prediction (for these studies, the signal represents a tracked objects state vector). For the special case of stationary statistics (a stochastic process whose joint probability distribution does not change when passing through time or space), Weiner showed how to predict random variables and separate random signals from random noise through the Weiner-Hopf integral equation. This equation is based on the concept that a random observed signal $F(t)$ is bisected into two distinct parts, the actual signal $G(t)$, and some random noise $[F(t) - G(t)]$, which along with the application of an impulse filter response, is used to obtain an estimate for the actual signal. This governing equation, as well as the Weiner-Hopf convolution integral used to solve this equation can be seen in Eqs. 3.6 and 3.7 respectively.

$$F(t) = G(t) + [F(t) - G(t)] \tag{3.6}$$

$$\chi(t+h) - \int_0^\infty \phi(t-\tau)\mathcal{K}(\tau)d\tau = 0 \tag{3.7}$$

where the auto correlation of $F(t)$, and the cross correlation of $F(t)$ and $G(t)$ are given by ϕ and χ respectively. Equation 3.7 is used to solve for an operator $\mathcal{K}(\tau)$ which can be used to solve the minimum error prediction estimate for $G(t+h)(h >$

0) when applied to the equation

$$G(t+h) = \int_0^\infty F(t-\tau)\mathcal{K}(\tau)d\tau \tag{3.8}$$

Therefore, the actual signal $G(t)$ can be solved for based simply on statistical properties (those being auto and cross correlations) of the corrupted signal, and not on knowing the actual signal itself. This framework provides the basis for applying a "filter" to a signal, which can be viewed generally in graphical format in Figure 3.5.

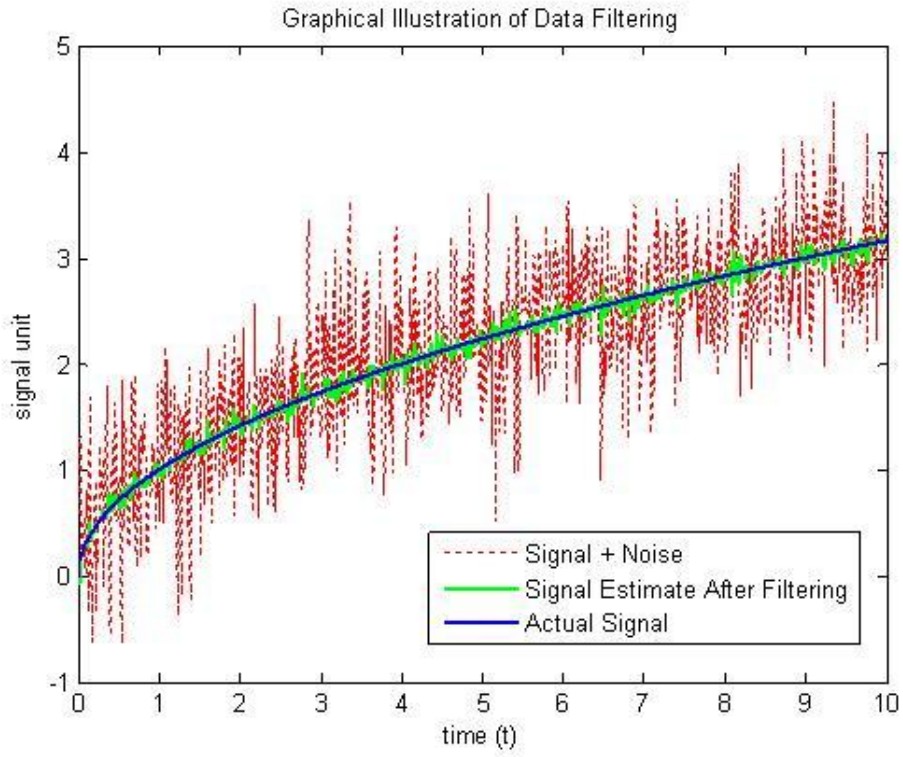

Figure 3.5. Illustration of a signal (blue), that same signal corrupted by noise (red dashed), and the signal estimate which can be gained through signal observations and filtering (green)

However, the solution to the integral in Equation 3.7 is not trivial, and has some associated problems. Two of the issues that make this solution difficult are

1) Numerical determination and statistics required to obtain the impulse response is quite involved and difficult for computers to handle (at the time

Weiner's paper was published)

2) Various generalizations to the filtering problem, such as growing-memory filters and non-stationary prediction can require difficult derivations for each particular case

3) Mathematics in the derivations of the Wiener-Hopf equation tend to have obscurities in the fundamental assumptions of the problem and how they effect the final solution.

Due to these shortcomings, Kalman decided to invent his own process for simplifying the mathematics in the Wiener-Hopf equation, which would eventually become the Kalman Filter. First, Kalman noticed that the Wiener problem requires the use of distributions and expected values, for which first and second order averages could be used as approximations to these values, making 2) more clear. Next, Kalman, used the work of Bode and Shannon [45] to represent arbitrary random signals as an output of some linear system which is perturbed by some uncorrelated random "white noise". In particular, this method of viewing a signal allowed for a solution to the Wiener problem using a state transition matrix. This approach was applicable in many generalizations of the filtering problem, solving the complexities of 1). Eventually, this would lead to a differential equation for the covariance matrix of the optimal estimation error, in a similar way as the Wiener-Hopf equation (which while not explicitly presented above, finding a signal error and computing the norm is all that is needed to obtain the covariance). This covariance matrix can be calculated at an initial time which an observation is made and at each time of subsequent observations so that the coefficients for an optimal linear filter can be obtained.

3.2.2 The Linear Kalman Filter

The general framework for the formulation of the linear Kalman filter is that of a stochastic system based upon a given signal $x_{1,k}$ and noise $x_{2,k}$ at a given time $t = k\Delta t$. These two quantities can be combined to create an observation y_k at this time, such that

$$y_k = x_{1,k} + x_{2,k} \tag{3.9}$$

Given an observation, it is necessary to filter out the noise from the actual signal, allowing for an estimation of the signal at a given time (the same goal as Equation 3.8, but accomplished now by different means). Once the filtering process is complete, the estimate can be used to aid in various real-time performance/information metrics while observations are taking place, as well as the creation of a predictive model of the system. It should be noted that in this case $x_{1,k}$, $x_{2,k}$ and y_k are considered to be random processes.

Additionally, the random variable $x_{1,k}$ can be estimated statistically to obtain the signal estimate at a given time $\hat{X}_{1,k}$, which will differ from the actual signal value by an estimation error $\epsilon = x_{1,k} - \hat{X}_{1,k}$. This estimation error can then be implemented in a loss function $L(\epsilon)$ which as long as it is always a positive and a non-decreasing function of the estimation error, can be used as a cost function in a minimization problem. Thus, an optimal filter should (but is not required to) choose an estimation $\hat{X}_{1,k}$ which minimizes the expected value of the loss function $\mathbb{E}[L(\epsilon)]$.

Since the estimation error will be a function of a set of observations $y_{0,k}, ..., y_{n,k}$, it follows that the solution for the optimal estimate will be a function of these observed variables. Obtaining a general solution to this problem for all practical purposes is impossible, unless one of the following is true

i) The random processes $x_{1,k}$ and $x_{2,k}$ are Gaussian

ii) The optimal estimate can only be a linear function the random observation variables and the loss function $L(\epsilon) = \epsilon^2$

If one of these two conditions are met, it can be shown that given the random observation variables $y_{0,k}, ..., y_{n,k}$, a vector space $\bar{Y}_k$ can be created where each point or vector in $\bar{Y}_k$ can be defined by a linear combination of $y_{0,k}, ..., y_{n,k}$ with real coefficients (for example, each vector could be defined as such $\sum_{i=0}^{k} a_i y_i$). Through some derivation, it can be shown that the optimal signal estimate at some time $k+1$, which is expressed as $\hat{X}^*_{1,k+1}$ (that is, the signal estimate which minimizes the loss function $L(\epsilon)$) is the orthogonal projection of $x_{1,k+1}$ (assumed to be a random

variable) on the vector space $\bar{Y}_{k+1}$. Thus, the random variable $x_{1,k+1}$ can be broken down to components orthogonal to the vector space $\bar{Y}_{k+1}$, and components within the vector space $\bar{Y}_{k+1}$, the latter representing the orthogonal projection of $x_{1,k+1}$ on $\bar{Y}_{k+1}$. This optimal estimate represents an output for an open loop impulse filtering process, with the input being the set of random observation variables [46],[47].

Now that a general solution for an impulse filter is established, it is necessary to then apply this solution to a time-varying signal. This process requires both propagating the system dynamics through time, and solving for the optimal estimate at each observation. Additionally, since the random variable $x_{1,k+1}$ is never known, but its estimate at the previous time step is, $\hat{X}_{1,k}$ must be propagated through time Δt so that an approximation can be made for $x_{1,k+1}$, allowing to solve for the orthogonal projection.

As for the governing dynamics of the system, since the signal represents a random process, it can be broken down into a process governed by known dynamics which is excited by an independent random process. In this system, it is also assumed that the independent random process is Gaussian, which allows for not only the previous derivation of the optimal signal estimate, but also is helpful because this random noise will remain Gaussian when passing through any linear system. In these studies, this linear system is taken to be the propagation of the signal (in the wording of dynamic systems, this signal can also be viewed as the state) from one given initial time to some final desired time, by means of implementing a state transition matrix Φ. The state transition matrix represents an $n \times n$ matrix (where n is the number of state variables) in which each element in the matrix can be determined by solving a first order differential equation. Given a linear dynamic system, and linear observation functions expressed in the form (with the addition of the subscript object index i as was used previously in Chapter 2)

$$\frac{d\vec{x}_{i,k}}{dt} = F_{i,k}\vec{x}_{i,k} + D_{i,k}\vec{u}_{i,k} \tag{3.10}$$

$$\vec{y}_{i,k} = M_{i,k}\vec{x}_{i,k} \tag{3.11}$$

where $\vec{x}_{i,k}$ is the $n \times 1$ state vector of the system, $\vec{u}_{i,k}$ is an $m \times 1$ ($m \leq n$) vector of system inputs (which for these studies is similar to the process noise found in Eq. 2.2), and $\vec{y}_{i,k}$ is a $p \times 1$ vector representing the system outputs. The matrices $F_{i,k}$, $D_{i,k}$, and $M_{i,k}$ can be either time-varying or stationary. This method of describing an objects dynamics is the linear version of Equations 2.2 and 2.15 respectively. The following equation for the propagation of the state can be rewritten using the nomenclature for the state transition matrix, and assuming that the input $\vec{u}_{i,k}$ is a zero mean Gaussian distribution independent of the state, results in

$$\vec{x}_{i,k+1} = \Phi_{i,k|k+1}\vec{x}_{i,k} + \vec{u}_{i,k} \tag{3.12}$$

A description of solving for the elements of the state transition matrix $\Phi_{i,k|k+1}$ can be found in Chapter 2. Using the above linear dynamics, and taking the loss function to be the trace of the covariance of the optimal estimate

$$L(\epsilon) = \epsilon^2 = tr\left(\mathbb{E}\left[\vec{x}_{i,k} - \hat{X}^*_{i,k}\right]\left[\vec{x}_{i,k} - \hat{X}^*_{i,k}\right]^T\right) = tr\left(\hat{P}^*_{i,k}\right) \tag{3.13}$$

the goal of the Kalman Filter algorithm then becomes to find the optimal state estimate for the next time step $\hat{X}^*_{i,k+1}$ which will minimize the trace of the covariance matrix of the estimation error $\hat{P}^*_{i,k+1}$. Once the closed loop process iterates another time step, the counter k iterates one unit, and $\hat{X}^*_{i,k+1}$ will become $\hat{X}^*_{i,k}$ for the current iteration.

The optimal state estimate $\hat{X}^*_{i,k+1}$ is solved by applying the orthogonal projection described previously, in conjunction with propagating the (assumed) linear dynamics through the state transition matrix, thus creating a closed loop system which will filter noise from the observations, successfully finding the signal (or state) within the addition of Gaussian noise.

Mathematically, Kalman showed that the optimal state estimate $\hat{X}^*_{i,k+1}$ is calculated from the following formula

$$\hat{X}^*_{i,k+1} = \Phi_{i,k|k+1}\hat{X}^*_{i,k} - \Phi_{i,k|k+1}\hat{P}^*_{i,k}M^T_{i,k+1}\left[M_{i,k+1}\hat{P}^*_{i,k}M^T_{i,k+1}\right]^{-1}\left(M_{i,k+1}\hat{X}^*_{i,k} - \vec{y}_{i,k+1}\right) \tag{3.14}$$

Additionally, a recursion relation can be formulated to propagate the covariance forward in time, and is given by

$$\hat{P}^*_{i,k+1} = \Phi_{i,k|k+1}\left\{\hat{P}^*_{i,k} - \hat{P}^*_{i,k}M^T_{i,k+1}\left[M_{i,k+1}\hat{P}^*_{i,k}M^T_{i,k+1}\right]^{-1}\hat{P}^*_{i,k}M_{i,k+1}\right\}\Phi^T_{i,k|k+1} + \Upsilon_{i,k} \tag{3.15}$$

where $\Upsilon_{i,k}$ is the covariance of the independent random Gaussian noise input, such that $\Upsilon_{i,k} = \vec{u}_{i,k}\vec{u}^T_{i,k}$. Therefore, given an initial estimate of the covariance $\hat{P}^*_{i,0}$, an estimate of the state $\hat{X}^*_{i,0}$, as well as a covariance of the random process noise, a closed loop algorithm can be used to solve for the optimal state and covariance estimates at discrete time intervals given the observations $y_{(i,1),k}, ..., y_{(i,p),k}$, thus separating the signal (or state) from the independent random zero mean Gaussian noise. This effectively creates an iterative process for solving the Wiener equation. In Kalman's formulation, $\hat{X}^*_{i,k+1}$ is the estimate for the signal (represented by $G(t)$ in Eqs. 3.6 and 3.7) and the diagonals of the covariance estimate $\hat{P}^*_{i,k+1}$ represent variances (i.e. a characteristic of noise in the form of uncertainty) for each state variable. In addition, the above formulation for updating state and covariance estimates is based on a batch process, where these estimates are updated in one calculation step. Further implementations of this Kalman filter algorithm (such as the extended Kalman filter, and unscented Kalman filter) use a sequential process, where estimates are updated using multiple calculation steps. Equations 3.14 - 3.15 can be broken into a sequential process where an initial forecast step propagates estimates forward in time to yield

$$\hat{X}^f_{i,k+1} = \Phi_{i,k|k+1}\hat{X}^*_{i,k} \tag{3.16}$$

and

$$\hat{P}^f_{i,k+1} = \Phi_{i,k|k+1}\hat{P}^*_{i,k}\Phi^T_{i,k|k+1} \tag{3.17}$$

If these estimates from Eqs. 3.16 - 3.17, as well as the definition of the cross covariance $\left(P^{xy}_{i,k+1} = P^{f}_{i,k+1}M^{T}_{i,k+1}\right)$, innovation covariance $\left(P^{yy}_{i,k+1} = M_{i,k+1}P^{f}_{i,k+1}M^{T}_{i,k+1}\right)$, and Kalman gain $\left(K_{i,k+1} = P^{xy}_{i,k+1}\left[P^{yy}_{i,k+1}\right]^{-1}\right)$ are substituted into Eqs. 3.14 - 3.15, the result is

$$\hat{X}^{*}_{i,k+1} = \hat{X}^{f}_{i,k+1} + K_{i,k+1}\left[\vec{y}_{i,k+1} - M_{i,k+1}\hat{X}^{f}_{i,k+1}\right] \tag{3.18}$$

and

$$\hat{P}^{*}_{i,k+1} = \hat{P}^{f}_{i,k+1} - K_{i,k+1}P^{yy}_{i,k+1}K^{T}_{i,k+1} \tag{3.19}$$

In further implementations of this basic Kalman filter algorithm, Eqs. 3.18 - 3.19 are modified to handle nonlinear systems (as well as the inclusion of sensor noise), and are found in Section 3.3 for the extended Kalman filter, and Section 3.4 for the unscented Kalman filter.

3.3 The Extended Kalman Filter

One important attribute of the linear Kalman filter is that optimal estimates are based on a linear system, with linear measurements with respect to the state vector. However, for the case of a nonlinear system (such as the governing equations of satellite motion), the linear Kalman filter can actually be reformatted using several methods to account for these nonlinearities. One of the oldest and most popular methods for applying the Kalman filter algorithm to nonlinear systems is the extended Kalman filter (EKF), whose inception took place shortly after Kalman's first paper [23]. The EKF algorithm differs from that of the original linear Kalman filter primarily in the way state estimates are propagated forward in time, and how a linear approximation is made to a nonlinear measurement function. This linear approximation is carried out fairly simply, in which the linear measurement matrix $M_{i,k+1}$ is approximated as the Jacobian of the nonlinear measurement function evaluated at the current object estimate, such that

$$M_{i,k+1} \approx H_{i,k+1} = \left[\frac{\partial h(\vec{x})}{\partial \vec{x}}\right]_{\vec{x}=\hat{X}^f_{i,k+1}} \tag{3.20}$$

where the superscript f represents the state estimate propagated forward in time from time step k to time step $k+1$ (discussed later in Chapter 3, section 3.1). It should be noted that $H_{i,k+1}$ is a first order approximation of the nonlinear measurement function, and therefore may be subject to inaccuracies from neglecting higher order Taylor series terms, an aspect of the EKF which may lead to problems when applied to highly nonlinear systems [27].

Unlike the linear Kalman filter, which used a batch process to propagate the state estimate using a state transition matrix simultaneously with calculating the optimal estimate of the current time step, the EKF established a sequential process in which the state and covariance estimates are propagated first in a *forecast step*, and then the optimal state and covariance estimates are calculated based on observations $\vec{y}_{i,k+1}$ in an *update step*. The primary reason for creating this sequential process is that unlike the linear Kalman filter, the EKF handles propagates state estimates by direct integration of an object's nonlinear equations of motion (for these studies, the equations of motion are presented in Eq. 2.2). This results in a nonlinear transformation between time $t = k\Delta t$ and $t = (k+1)\Delta t$, and therefore a batch process like in Eqs. 3.14 and 3.15 cannot be applied. A general algorithm to the EKF has been well established [25, 48], and has also been applied to tracking problems involving orbit determination, similar to the SSA problem [23, 29, 28, 21, 49].

3.3.1 Forecast Step

Taking object dynamics and measurement to be nonlinear, and therefore described by Equations 2.2 and 2.15, it follows that the forecast step be carried out by propagating both the state and covariance estimates to achieve the forecast state and covariance estimates given in Equations 3.21 and 3.22 respectively. In this step the EKF takes an initial estimate the object's state $\hat{X}^*_{i,k}$ and covariance $\hat{P}^*_{i,k}$ and propagates them from $t = k\Delta t$ to $t = (k+1)\Delta t$ to obtain the forecast estimates

for each,

$$\hat{X}^f_{i,k+1} = \mathcal{F}(\hat{X}^*_{i,k}) \tag{3.21}$$

$$\hat{P}^f_{i,k+1} = \Phi_{i,k|k+1}\hat{P}^*_{i,k}\left[\Phi_{i,k|k+1}\right]^T + Q_{i,k} \tag{3.22}$$

where $\mathcal{F}(\cdot)$ represents the numerical integration of the nonlinear function $f(\cdot)$ from $t = [k\Delta t, (k+1)\Delta t]$. Of particular note is the fact that the covariance is still propagated through the linear state transition matrix while the state estimate is propagated nonlinearly by means of numerical integration. This linear propagation of the covariance estimate is one facet of the EKF which could possibly lead to errors in estimating the covariance should the system be highly nonlinear.

3.3.2 Update Step

In the update step, state and covariance estimates are updated for all objects tasked to be observed (methods for determining objects tasked for observation in the set $\mathcal{O}^\tau_{k+1}$, as well as the sets of sensors tasked to observed them $\mathcal{S}^\tau_{i,k+1}$ are detailed in Chapter 4). Measurement data from all M' ($M' \leq M$) sensors tasked to observe object i at time step $k+1$ in the set $\mathcal{S}^\tau_{i,k+1} = \{s^\tau_1, \ldots, s^\tau_{M'}\}$ (where elements in $\mathcal{S}^\tau_{i,k+1}$ represent a sensor index j) is given by the vector

$$\vec{y}_{i,k+1} = \begin{bmatrix} h_\rho\left(\vec{x}_{i,k+1}, \vec{s}_{s^\tau_1,k+1}\right) + v^\rho_{s^\tau_1,k+1} \\ \vdots \\ h_\rho\left(\vec{x}_{i,k+1}, \vec{s}_{s^\tau_{M'},k+1}\right) + v^\rho_{s^\tau_{M'},k+1} \\ h_\psi\left(\vec{x}_{i,k+1}, \vec{s}_{s^\tau_1,k+1}\right) + v^\psi_{s^\tau_1,k+1} \\ \vdots \\ h_\psi\left(\vec{x}_{i,k+1}, \vec{s}_{s^\tau_{M'},k+1}\right) + v^\psi_{s^\tau_{M'},k+1} \end{bmatrix} \tag{3.23}$$

Next, optimal state estimates are obtained, first by calculating an estimated measurement vector

$$\hat{y}_{i,k+1} = \begin{bmatrix} h_\rho\left(\hat{X}^f_{i,k+1}, \vec{s}_{s^\tau_1,k+1}\right) \\ \vdots \\ h_\rho\left(\hat{X}^f_{i,k+1}, \vec{s}_{s^\tau_{M'},k+1}\right) \\ h_\psi\left(\hat{X}^f_{i,k+1}, \vec{s}_{s^\tau_1,k+1}\right) \\ \vdots \\ h_\psi\left(\hat{X}^f_{i,k+1}, \vec{s}_{s^\tau_{M'},k+1}\right) \end{bmatrix} \tag{3.24}$$

along with the *innovation covariance* as well as the *cross covariance*, calculated respectively by

$$P^{yy}_{i,k+1} = H_{i,k+1}\hat{P}^f_{i,k}H^T_{i,k+1} \tag{3.25}$$

$$P^{xy}_{i,k+1} = \hat{P}^f_{i,k+1}\left[H_{i,k+1}\right]^T \tag{3.26}$$

where

$$H_{i,k+1} = \begin{bmatrix} \frac{\partial}{\partial\vec{x}}h_\rho\left(\vec{x}, \vec{s}_{s^\tau_1,k+1}\right) \\ \vdots \\ \frac{\partial}{\partial\vec{x}}h_\rho\left(\vec{x}, \vec{s}_{s^\tau_{M'},k+1}\right) \\ \frac{\partial}{\partial\vec{x}}h_\psi\left(\vec{x}, \vec{s}_{s^\tau_1,k+1}\right) \\ \vdots \\ \frac{\partial}{\partial\vec{x}}h_\psi\left(\vec{x}, \vec{s}_{s^\tau_{M'},k+1}\right) \end{bmatrix}_{\vec{x}=\hat{X}^f_{i,k+1}} \tag{3.27}$$

These are used to calculate the *Kalman gain matrix*, which is defined as an optimal operator that minimizes the trace of the estimated covariance $\hat{P}^*_{i,k+1}$ [25]. In the linear Kalman filter, the same minimization is solved for, but Eqs. 3.14 and 3.15 do not include a specific calculation for the Kalman gain matrix. However, through minimal derivation it can be shown that the definition of $K_{i,k+1}$ in the EKF can also be found in the linear case, but is not defined explicitly in Kalman's original formulation. The Kalman gain matrix is calculated from

$$K_{i,k+1} = P^{xy}_{i,k+1}\left\{P^{yy}_{i,k+1} + R^\tau_{i,k+1}\right\}^{-1} \tag{3.28}$$

where the measurement noise covariance $R^{\tau}_{i,k+1}$ for object i based on the sensors in the set $\mathcal{S}^{\tau}_{i,k+1}$ tasked to observe it is calculated by

$$R^{\tau}_{i,k+1} = \begin{bmatrix} \left(v^{\rho}_{s^{\tau}_1,k+1}\right)^2 & 0 & \cdots & & \cdots & 0 \\ 0 & \ddots & & & & \vdots \\ \vdots & & \left(v^{\rho}_{s^{\tau}_{M'},k+1}\right)^2 & & & \\ \vdots & & & \left(v^{\psi}_{s^{\tau}_1,k+1}\right)^2 & & \vdots \\ \vdots & & & & \ddots & 0 \\ 0 & \cdots & & \cdots & 0 & \left(v^{\psi}_{s^{\tau}_{M'},k+1}\right)^2 \end{bmatrix} \tag{3.29}$$

Using these variables, the state and covariance estimates are updated using the equations

$$\hat{X}^{*}_{i,k+1} = \hat{X}^{f}_{i,k+1} + K_{i,k+1}[\vec{y}_{i,k+1} - \hat{y}_{i,k+1}] \tag{3.30}$$

$$\hat{P}^{*}_{i,k+1} = \hat{P}^{f}_{i,k+1} - K_{i,k+1}\left[P^{yy}_{i,k+1} + R^{\tau}_{i,k+1}\right] K^{T}_{i,k+1} \tag{3.31}$$

It should be noted that if the linear Kalman filter is broken up into a sequential process like the EKF, Eqs. 3.30 and 3.31 are the same as 3.14 and 3.15, except that $M_{i,k+1}$ is replaced by $H_{i,k+1}$ and the propagation of the state estimate changes from linear (using $\Phi_{i,k|k+1}$) to nonlinear (using $\mathcal{F}$).

An algorithm which describes the step-by-step process of implementing the EKF for an object i (assuming measurement data exists for all M_s sensors at each time step) can be found in Appendix B.

3.4 The Unscented Kalman Filter

For highly nonlinear systems, or those in which first order accuracy is not good enough to properly describe the system, another application of the Kalman filter algorithm exists which makes no linear approximations to the dynamics or mea-

surement functions. This algorithm, originally proposed by Julier and Uhlmann in 2004 [27] is known as the Unscented Kalman Filter (UKF) and can provide a more robust filter than the EKF for certain nonlinear systems. Unlike the EKF, the UKF creates a distribution of *sigma points* around the state estimate, and propagates them through the nonlinear dynamics and measurement functions directly. This propagation of sigma points is then applied to statistical analysis to directly calculate various parameters in the Kalman filter algorithm, leading to second order or higher accuracy in the approximation of the nonlinearities of the system [27]. This nonlinear propagation, followed by direct calculation of the mean and covariance based on the sigma point distribution encompasses the essence of the unscented transformation used in the UKF, and can be seen in a general sense in Figure 3.6.

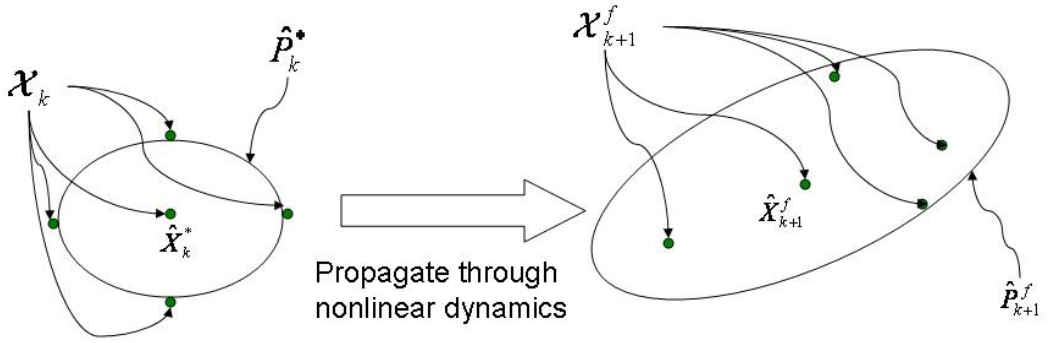

Figure 3.6. Illustration of propagation of sigma points through nonlinear object equations of motion to calculate forecast state and covariance estimates. The ellipses represent a 3σ Gaussian error ellipse, which can be obtained from the covariance estimate $\hat{P}_k^*$ and $\hat{P}_{k+1}^f$

3.4.1 Initialization

The following is a description of applying the UKF, subject to purely additive zero mean Gaussian process and measurement noise, as presented in [1]. Given an initial object state estimate, $\hat{X}_{i,0}^*$ and covariance estimate $\hat{P}_{i,0}^*$ at time $t = 0$, an initial set of sigma points $\mathcal{X}_{i,0}$ are populated

$$\mathcal{X}_{i,0} = \hat{X}_{i,0}^* \left[1\right]_{1\times(2n+1)} + \sqrt{n+\Lambda}\left[0_{n\times 1}\ \hat{P}_{i,0}^{CH}\ -\hat{P}_{i,0}^{CH}\right] \tag{3.32}$$

where $\hat{P}_{i,0}^{CH}$ represents the Choleski factorization of $\hat{P}_{i,0}^{*}$ such that

$$\hat{P}_{i,0}^{CH}\left[\hat{P}_{i,0}^{CH}\right]^{T} = \hat{P}_{i,0}^{*} \tag{3.33}$$

and Λ is a scaling parameter, defined by $\Lambda = \alpha^2(n+\kappa) - n$ [28]. The constant α is generally small, and is chosen between the range [0,1], while the constant κ is used for tuning of higher order moments of the covariance ($\kappa = 3-n$ for Gaussian distributions). Choices for these constant parameters can vary, whose origins and selection strategies are detailed in [27] and [1]. For these studies, $\alpha = 0.001$ and $\kappa = 3 - n$ are used based on suggestions in various works using a similar UKF algorithm for orbit determination problems [28, 29].

3.4.2 Forecast Step

For some time step k, the sigma points $\mathcal{X}_{i,k}$ are propagated through the nonlinear system dynamics from $t = k\Delta t$ to $t = (k+1)\Delta t$

$$\mathcal{X}_{i,k+1}^{f} = \mathcal{F}(\mathcal{X}_{i,k}) \tag{3.34}$$

Once the sigma points are propagated, the forecast step is obtained from taking a weighted average of sigma points

$$\hat{X}_{i,k+1}^{f} = \sum_{\gamma=0}^{2n} W_x^{\gamma}\mathcal{X}_{i,k+1}^{f,\gamma} \tag{3.35}$$

where the mean weights W_x^{γ} corresponding to each sigma point can be calculated by

$$W_x^{\gamma} = \begin{cases} \frac{\Lambda}{(n+\Lambda)}, & \gamma = 0 \\ \frac{1}{2(n+\Lambda)}, & \gamma = 1,\ldots,2n \end{cases} \tag{3.36}$$

and γ represents a sigma point in the $2n+1$ collection of sigma points for an object, represented by $\mathcal{X}_{i,k+1}^{f}$ (γ stands for a column in the $[n \times 2n+1]$ sigma point matrix represented by $\mathcal{X}_{i,k+1}^{f}$). Given the forecast state estimate in Eq. 3.36, the forecast covariance estimate is calculated similarly by

$$\hat{P}^{f}_{i,k+1} = \sum_{\gamma=0}^{2n} W_p^{\gamma} \left[\mathcal{X}^{f,\gamma}_{i,k+1} - \hat{X}^{f}_{i,k+1}\right] \left[\mathcal{X}^{f,\gamma}_{i,k+1} - \hat{X}^{f}_{i,k+1}\right]^T + Q_{i,k} \tag{3.37}$$

where the covariance weights W_p^{γ} corresponding to each sigma point can be calculated by

$$W_p^{\gamma} = \begin{cases} \frac{\Lambda}{(n+\Lambda)} + (1 - \alpha^2 + \beta), \ \gamma = 0 \\ \frac{1}{2(n+\Lambda)}, \ \gamma = 1, \ldots, 2n \end{cases} \tag{3.38}$$

where the constant β is another parameter used to help incorporate higher order effects, and is set at the optimal value for a Gaussian distribution, $\beta = 2$ [27]. It should also be noted that in cases where the process noise and measurement noise are not purely additive, separate sigma points will need to be added in accordance with [27] and [29].

3.4.3 Update Step

In the update step, state and covariance estimates are updated for all objects tasked to be observed (methods for determining objects tasked for observation in the set $\mathcal{O}^{\tau}_{k+1}$, as well as the sets of sensors tasked to observed them $\mathcal{S}^{\tau}_{i,k+1}$ are detailed in Chapter 4). Measurement data from all M' ($M' \leq M$) sensors tasked to observe object i at time step $k+1$ in the set $\mathcal{S}^{\tau}_{i,k+1} = \{s^{\tau}_1, \ldots, s^{\tau}_{M'}\}$ (where elements in $\mathcal{S}^{\tau}_{i,k+1}$ represent a sensor index j) is given by Eq. 3.23. In order to calculate the measurement estimate $\hat{y}_{i,k+1}$, the sigma points are inputed into the nonlinear measurement function such that

$$\mathcal{Y}^{\gamma}_{i,k+1} = \begin{bmatrix} h_{\rho}\left(\mathcal{X}^{f,\gamma}_{i,k+1}, \vec{s}_{s^{\tau}_1,k+1}\right) \\ \vdots \\ h_{\rho}\left(\mathcal{X}^{f,\gamma}_{i,k+1}, \vec{s}_{s^{\tau}_{M'},k+1}\right) \\ h_{\psi}\left(\mathcal{X}^{f,\gamma}_{i,k+1}, \vec{s}_{s^{\tau}_1,k+1}\right) \\ \vdots \\ h_{\psi}\left(\mathcal{X}^{f,\gamma}_{i,k+1}, \vec{s}_{s^{\tau}_{M'},k+1}\right) \end{bmatrix}, \ \gamma = 0, \ldots, 2n \tag{3.39}$$

The estimated measurement can then be calculated from

$$\hat{y}_{i,k+1} = \sum_{\gamma=0}^{2n} W_x^{\gamma} \mathcal{Y}_{i,k+1}^{\gamma} \tag{3.40}$$

It is again important to recognize that the UKF does not rely on a linear approximation of the Taylor series expansion of the measurement function about $\hat{X}_{i,k+1}^{f}$, as with the EKF, but instead is accurate to at least second order, or even third order for Gaussian inputs. A proof of this accuracy is provided in both [27] and [1].

With the necessary calculations to perform the update step, the cross covariance is calculated by

$$P_{i,k+1}^{xy} = \sum_{\gamma=0}^{2n} W_p^{\gamma} \left[\mathcal{X}_{i,k+1}^{f,\gamma} - \hat{X}_{i,k+1}^{f} \right] \left[\mathcal{Y}_{i,k+1}^{\gamma} - \hat{y}_{i,k+1} \right]^T \tag{3.41}$$

while the innovation covariance is calculated by

$$P_{i,k+1}^{yy} = \sum_{\gamma=0}^{2n} W_p^{\gamma} \left[\mathcal{Y}_{i,k+1}^{\gamma} - \hat{y}_{i,k+1} \right] \left[\mathcal{Y}_{i,k+1}^{\gamma} - \hat{y}_{i,k+1} \right]^T \tag{3.42}$$

All the necessary information to calculate the updated state and covariance estimates in Eqs. 3.30 and 3.31 is obtained, to which these calculations are carried out in the same manner as in the EKF and linear Kalman filter (should it be separated into a sequential process). Again, as with comparing the EKF to the linear Kalman filter, the UKF carries out the same calculation process as the EKF, it just handles propagating the state and covariance estimates, as well as approximating the measurement function differently.

3.4.4 Performance Discrepancies Between the UKF and EKF

Due to the method of using sigma points to calculate forecast estimates, measurement approximations, and other variables, many studies have shown the UKF to produce slightly better estimates when applied to nonlinear systems than the EKF

[28], [29]. It is expected that these benefits will also be present in the calculation of various tasking metrics, and the UKF should gain increased performance when compared to the EKF for multi-object tracking scenarios which employ dynamic sensor tasking.

One such illustration of these advantages is in how the UKF better handles the calculation of forecast step uncertainty (reflected by the covariance estimate) than the EKF for nonlinear systems. Using the same example in Section 3.1.1, Figures 3.7 and 3.8 show how both the EKF and UKF propagate the initial mean given in Eq. 3.4 with an initial Gaussian covariance derived from the distribution generated from standard deviations in Eq. 3.5 as seen in Figure 3.1. Figures 3.7 and 3.8 show that the result ing estimated uncertainty obtained from the UKF provide a better model for the overall uncertainty of the non-Gaussian distribution than the EKF, an advantage that can lead to better overall estimates, and as this research will illustrate, better tasking decisions.

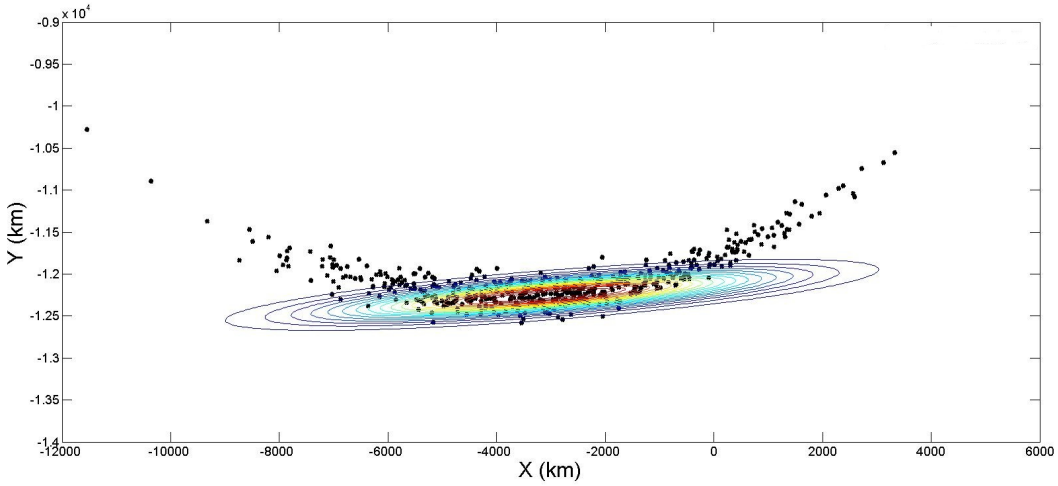

Figure 3.7. Propagation of initial Gaussian distribution showing contour overlay of the evaluation of Eq. 3.1 with EKF forecast mean and covariance.

An algorithm which describes the step-by-step process of implementing the UKF for an object i (assuming measurement data exists for all M_s sensors at each time step) can be found in Appendix B.

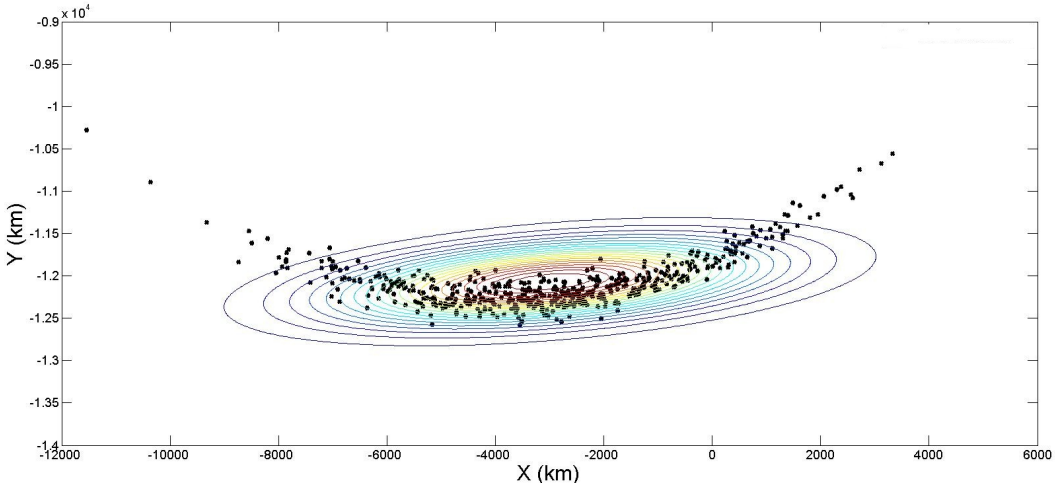

Figure 3.8. Propagation of initial Gaussian distribution showing contour overlay of the evaluation of Eq. 3.1 with UKF forecast mean and covariance.

3.5 Gaussian Mixture Models and the AEGIS Filter

Recently introduced by DeMars et. al. [35, 36, 2], the AEGIS filter reflects an adaptation to a GMM filtering strategy, in which a series of L weighted Gaussian PDF's is used within a UKF filtering scheme to obtain state and covariance estimates. The main conjecture of this method is that the use of a GMM to describe the overall PDF is advantageous when applied to non-Gaussian PDF's (such as the case with orbit determination problems). In addition, the AEGIS filter employs a unique method for determining when a single Gaussian PDF must be split into multiple ones based on a calculation of entropy, which reflects the presence of nonlinearity in a PDF. As follows are details of both GMM filters, and how they are implemented into the AEGIS filtering strategy.

3.5.1 The Gaussian Mixture Model Distribution

As illustrated by the work of Sorenson et. al. [31], GMM's are shown to be extremely helpful because they can use the easily characterized Gaussian PDF to approximate a large class of PDF's. This is achieved by using several Gaussian

PDF's to approximate a single non-Gaussian PDF, therefore more accurately reflecting a PDF result ing from propagating an initially Gaussian PDF through a nonlinear system. In the GMM case, the overall PDF of the distribution is described by the summation (presented to reflect and object i and time step k used in these studies)

$$p^{gmm}\left(\vec{x}_{i,k};\hat{X}^{*}_{i,k},\hat{P}^{*}_{i,k}\right)=\sum_{l=1}^{L}\nu^{l}_{i,k}p^{g}_{i,k}\left(\vec{x}_{i,k};\hat{X}^{*,l}_{i,k},\hat{P}^{*,l}_{i,k}\right) \tag{3.43}$$

where the index l represents a single component of the GMM containing L total Gaussian PDF's, and $\nu^{l}_{i,k}$ is a weight for the l^{th} component of the GMM for object i at time step k subject to the constraints

$$\nu^{l}_{i,k}\geq 0\ \forall\ l\in\{1,2,\ldots,L\}\ \text{ and }\ \sum_{l=1}^{L}\nu^{l}_{i,k}=1 \tag{3.44}$$

Sorenson also illustrated that as the number of components in the GMM increases, the hyper volume of their individual covariances decreases, leading eventually to each component reflecting an impulse function. The result is that given an unbounded number of components, a GMM can characterize a large number of non-Gaussian PDF's while still maintaining the convenient Gaussian property of each component. This fact makes the use of GMM's very convenient when used within an estimation process for a nonlinear system driven by some form of Kalman filtering scheme.

3.5.2 Splitting a Gaussian Distribution

Of critical importance in the implementation of a filtering scheme using a GMM (and therefore implementation of the AEGIS filter) is determining how to split a single Gaussian distribution into several Gaussian PDF's. The following is a brief synopsis of the methods used by DeMars [2] which concerns methods of splitting a multivariate[4] Gaussian distribution into one or more smaller Gaussian distributions. Therefore, in this case the assumption is that the L total means and

[4]A multivariate Gaussian distribution is described by a mean and covariance, while a univariate is described by a mean and variance

covariances in the GMM can be summated subject to a weight assigned to each component to yield an approximation of a single Gaussian PDF

$$p^g\left(\vec{x}_{i,k}; \hat{X}^*_{i,k}, \hat{P}^*_{i,k}\right) \approx \sum_{l=1}^{L} \nu^l_{i,k} p^g\left(\vec{x}_{i,k}; \hat{X}^{*,l}_{i,k}, \hat{P}^{*,l}_{i,k}\right) \tag{3.45}$$

Several recent works have illustrated methods for splitting both univariate [50, 51] and multivariate [52, 53] Gaussian distributions into two or more element GMM's. In the AEGIS filter, DeMars builds on methods first presented by Huber et. al. [54] posing a splitting method for multivariate distributions based on univariate splitting techniques projected along a single direction of the multivariate distribution. Therefore, an understanding of the splitting of a univariate distribution must first be presented before the method for splitting a multivariate distribution can be explained.

In the univariate case, a splitting technique is derived that can split a standard Gaussian distribution into multiple components. This is achievable due to the convenient property that any Gaussian distribution can be created by applying a linear transformation to the standard Gaussian distribution [2]. The standard univariate Gaussian distribution, which has zero mean and a standard deviation of one can be described by the PDF (dropping the subscripts i and k to reflect a general case, and not specific to the SSA problem in these studies)

$$p_1(x) = p^g(x; 0, 1) \tag{3.46}$$

where x is the single random state variable in this general example used to calculate the PDF. The goal of splitting $p_1(x)$ is to come up with a GMM $p_2(x)$ which approximates $p_1(x)$ such that

$$p_2(x) \approx \sum_{\tilde{l}=1}^{\tilde{L}} \nu^{\tilde{l}} p^g\left(x; m^{\tilde{l}}, \tilde{\sigma}^2\right) \tag{3.47}$$

where $m^{\tilde{l}}$ is the $\tilde{l}^{th}$ component mean and $\tilde{\sigma}^2$ is the variance of each of the $\tilde{l}$ components of the GMM. DeMars suggests that each standard deviation $\tilde{\sigma}$ be equal, due to problems that may arise from some being too large and others too

small should this constraint not exist. Therefore, to gain the best approximation of $p_2 \approx p_1$ it is necessary to form a minimization problem which finds the elements $m^{\tilde{l}}$, $\tilde{\sigma}$, and $w^{\tilde{l}}$ which results in both the smallest value of $\tilde{\sigma}^2$ and ensures a minimal distance between the p_1 and p_2 distributions. To achieve this, DeMars suggests a cost function of

$$J = \mathcal{D}\left(p_1, p_2\right) + \zeta\sigma^2 \tag{3.48}$$

where ζ is a scale factor to aid in the multi objective minimization of Eq. 3.48 and $\mathcal{D}\left(p_1, p_2\right)$ is the L_2 distance between the two distributions defined as [2][5]

$$\begin{aligned}\mathcal{D}\left(p_1, p_2\right) = & \sum_{a=1}^{L_1}\sum_{b=1}^{L_1}\nu_1^a\nu_1^b\mathcal{K}\left(\vec{m}_1^a, \vec{m}_1^b, P_1^a, P_1^b\right) + \ldots \\ & \ldots + \sum_{a=1}^{L_2}\sum_{b=1}^{L_2}\nu_2^a\nu_2^b\mathcal{K}\left(\vec{m}_2^a, \vec{m}_2^b, P_2^a, P_2^b\right) - \ldots \\ & \ldots - 2\sum_{a=1}^{L_1}\sum_{b=1}^{L_2}\nu_1^a\nu_2^b\mathcal{K}\left(\vec{m}_1^a, \vec{m}_2^b, P_1^a, P_2^b\right)\end{aligned} \tag{3.49}$$

where $\mathcal{K}\left(\cdot\right)$ is derived from DeMars and Maybeck [55] as

$$\begin{aligned}\mathcal{K}\left(\vec{m}_1, \vec{m}_2, P_1, P_2\right) = & \left|2\pi\left(P_1 + P_2\right)\right|^{-1/2} \times \ldots \\ & \ldots \times \exp\left\{-\tfrac{1}{2}\left(\vec{m}_1 - \vec{m}_2\right)^T\left(P_1 + P_2\right)^{-1}\left(\vec{m}_1 - \vec{m}_2\right)\right\}\end{aligned} \tag{3.50}$$

The minimization of Eq. 3.48 for a case of $\tilde{L} = 3$ has been executed by DeMars and will be used in the application of the AEGIS filter in these studies. Values for the cost function scalar value ζ, constant variance $\tilde{\sigma}^2$, and component means $m^{\tilde{l}}$ and weights $\nu^{\tilde{l}}$ for this case can be found in Table 3.1, and reflect values identical to those used in the work of DeMars.

Using the three component splitting library given in Table 3.1, a univariate splitting method can be applied to a multivariate GMM which approximates the PDF in Eq. 3.43. This is achieved by applying the univariate splitting method to each variable in the multivariate PDF, acting in a single direction of the mul-

[5] In this definition, p_1 and p_2 are both assumed to be GMM's of the form in Eq. 3.43. This form remains valid for the case of p_1 being a univariate standard Gaussian distribution and p_2 being a univariate GMM. In this case, $L_1 = 1$, $L_2 = \tilde{L}$, $\nu_1 = 1$, $\vec{m}_1 = 0$, and $P_1 = 1$ while for each $\tilde{l}$ component of p_2, $\vec{m}_2^{\tilde{l}} = m^{\tilde{l}}$, and $P_2^{\tilde{l}} = \tilde{\sigma}^2$.

Table 3.1. Weights $\left(\nu^{\tilde{l}}\right)$, means $\left(m^{\tilde{l}}\right)$, and variance $(\tilde{\sigma})$ for a $\tilde{L} = 3$ and $\zeta = 0.001$ solution to the minimization problem outlined in Eq. 3.48. These represent the splitting of an original Gaussian PDF to be approximated by three Gaussian PDF's with the corresponging weights and means, and all having equal variance.

$\tilde{l}$	$\nu^{\tilde{l}}$	$m^{\tilde{l}}$	$\tilde{\sigma}$
1	0.225224624913675	-1.05751546147588	0.671566288664076
2	0.54955075017265	0	0.671566288664076
3	0.225224624913675	1.05751546147588	0.671566288664076

tivariate PDF. Intuition would say that these directions should be determined by the principal axes of the covariance matrix (i.e. the eigenvectors of the covariance matrix), but aside from giving physical meaning to the splitting directions, this turns out to be unnecessary. In fact, a simple square root factor of the covariance matrix is all that is needed to determine a direction in which to apply a univariate splitting technique [2].

To illustrate this method, we start with an original weighted Gaussian PDF which is desired to be split into a GMM approximating the original PDF

$$\nu_{i,k}^{l} p^{g}\left(\vec{x}_{i,k}; \hat{X}_{i,k}^{*,l}, \hat{P}_{i,k}^{*,l}\right) \approx \sum_{\tilde{l}=1}^{\tilde{L}} \nu_{i,k}^{\tilde{l}} p^{g}\left(\vec{x}_{i,k}; \hat{X}_{i,k}^{*,\tilde{l}}, \hat{P}_{i,k}^{*,\tilde{l}}\right) \tag{3.51}$$

where the PDF on the left hand side of Eq. 3.51 represents a weighted Gaussian PDF before splitting, and the right hand side reflects a GMM after splitting. It should be noted that if the left hand side was to represent the original Gaussian PDF defined by the state and covariance estimates $p^{g}\left(\vec{x}_{i,k}; \hat{X}_{i,k}^{*}, \hat{P}_{i,k}^{*}\right)$ then Eq. 3.51 would be replaced by

$$\nu p^{g}\left(\vec{x}_{i,k}; \hat{X}_{i,k}^{*}, \hat{P}_{i,k}^{*}\right) \approx \sum_{\tilde{l}=1}^{\tilde{L}} \nu_{i,k}^{\tilde{l}} p^{g}\left(\vec{x}_{i,k}; \hat{X}_{i,k}^{*,\tilde{l}}, \hat{P}_{i,k}^{*,\tilde{l}}\right) \tag{3.52}$$

where $\nu = 1$. Using this definition, and calculating a square root factor (via Choleski factorization) for the covariance of the original PDF, $\hat{P}_{i,k}^{CH,l}\left[\hat{P}_{i,k}^{CH,l}\right]^{T} = \hat{P}_{i,k}^{*,l}$ where $\hat{P}_{i,k}^{CH,l} = \left[\hat{p}_{i,k}^{l,1}, \ldots, \hat{p}_{i,k}^{l,n}\right]$, the weights, means and covariances of the L' components in the GMM performed along the $\tilde{n}^{th}$ axis can be calculated from

$$\nu_{i,k}^{\tilde{l}} = \nu^{\tilde{l}} \nu_{i,k}^{l} \tag{3.53}$$

$$\hat{X}_{i,k}^{*,\tilde{l}} = \hat{X}_{i,k}^{*,l} + m^{\tilde{l}} \hat{p}_{i,k}^{l,\tilde{n}} \tag{3.54}$$

$$\hat{P}_{i,k}^{*,\tilde{l}} = \hat{P}_{i,k}^{CH,\tilde{l}} \left[\hat{P}_{i,k}^{CH,\tilde{l}} \right]^T \tag{3.55}$$

where $\hat{P}_{i,k}^{CH,\tilde{l}} = \left[\hat{p}_{i,k}^{l,1}, \ldots, \tilde{\sigma} \hat{p}_{i,k}^{l,\tilde{n}}, \ldots, \hat{p}_{i,k}^{l,n} \right]$. Again, it is important to note that should this split be the of the original Gaussian PDF and not of a component of a GMM, then Eqs. 3.53 - 3.55 would be replaced by

$$\nu_{i,k}^{\tilde{l}} = \nu^{\tilde{l}} \tag{3.56}$$

$$\hat{X}_{i,k}^{*,\tilde{l}} = \hat{X}_{i,k}^{*} + m^{\tilde{l}} \hat{p}_{i,k}^{\tilde{n}} \tag{3.57}$$

$$\hat{P}_{i,k}^{*,\tilde{l}} = \hat{P}_{i,k}^{CH,\tilde{l}} \left[\hat{P}_{i,k}^{CH,\tilde{l}} \right]^T \tag{3.58}$$

where $\hat{P}_{i,k}^{CH,\tilde{l}} = \left[\hat{p}_{i,k}^{1}, \ldots, \tilde{\sigma} \hat{p}_{i,k}^{\tilde{n}}, \ldots, \hat{p}_{i,k}^{n} \right]$, $\hat{P}_{i,k}^{CH} = \left[\hat{p}_{i,k}^{1}, \ldots, \hat{p}_{i,k}^{n} \right]$, and $\hat{P}_{i,k}^{CH} \left[\hat{P}_{i,k}^{CH} \right]^T = \hat{P}_{i,k}^{*}$.

3.5.3 Merging a GMM

The method of merging a GMM into a single Gaussian PDF is a necessary process in these studies, especially in the calculation of tasking metrics generated from AEGIS filter estimates, or assessing AEGIS performance. In some cases, the method of merging a GMM is necessary in a recursive GMM-based filtering process, especially if components of a GMM are redundant. In this case, some methods have been proposed which merge components based on their proximity calculated from a distance measure [56], while attempting to optimize a cost function determining components to merge result in in the minimum change between an original GMM and a reduced one [57].

In these studies, a primary application on the merging of GMM components is to determine the equivalent Gaussian PDF which replicates the GMM (an approximation reflected in Eq. 3.45). To calculate this, DeMars details how determining the expected values of each side of an equation similar to Eq. 3.45 and assuming all PDF's are Gaussian can yield approximations for a single mean and covariance of a GMM, given by

$$\hat{X}_{i,k}^{*,gmm} = \sum_{l=1}^{L} \frac{\nu_{i,k}^{l}}{\sum_{l=1}^{L} \nu_{i,k}^{l}} \hat{X}_{i,k}^{*,l} \tag{3.59}$$

and

$$\hat{P}_{i,k}^{*,gmm} = \sum_{l=1}^{L} \frac{\nu_{i,k}^{l}}{\sum_{l=1}^{L} \nu_{i,k}^{l}} \left(\hat{P}_{i,k}^{*,l} + \hat{X}_{i,k}^{*,l} \left[\hat{X}_{i,k}^{*,l} \right]^{T} \right) - \hat{X}_{i,k}^{*,gmm} \left[\hat{X}_{i,k}^{*,gmm} \right]^{T} \tag{3.60}$$

It should be noted that in Eq. 3.60, this weighted average covariance must take into account how far each GMM component mean (i.e. the l^{th} component) is away from the average weighted mean (given in Eq. 3.59), which is the reason why $\hat{X}_{i,k}^{*,l}$ and $\hat{X}_{i,k}^{*,gmm}$ are included in this equation [2]. Furthermore, an approximate single Gaussian mean and covariance can be calculated during anytime within a recursive GMM filtering strategy, such as in the forecast step, or once updates are obtained.

3.5.4 Detecting Nonlinearity in Propagation of a Gaussian Distribution

Given the nonlinear orbital dynamics governing the objects tracked in the SSA problem, a single Gaussian distribution may not provide an accurate approximation to the non-Gaussian error propagation, as illustrated in Section 3.1. However, should the degree of nonlinearity be small, a single Gaussian distribution may provide a suitable approximation to the error distribution, negating the need to split a Gaussian PDF into a GMM. Therefore, determining when these splitting events

occur should be based on the degree of nonlinearity in the propagation of a covariance estimate. Junkins et. al. [58] investigated the error propagation in orbital mechanics (using a two-body force model with J_2 and atmospheric drag perturbations), and discovered that the effects of nonlinearity in the error propagation are greatly affected by the coordinate system chosen. In this case, nonlinearity was measured by utilizing linear approximations of the orbital dynamics (i.e. a state transition matrix) to obtain a scalar quantification of nonlinearity called a *nonlinearity index*. Others such as Park et. al. [59] have built on this concept of using the state transition matrix to determine nonlinearity, but in this case higher order terms were included to use instead a state transition tensor to detect nonlinearity.

The AEGIS filter proposes a new method for determining nonlinearity, by utilizing the Jacobian of the nonlinear orbital dynamics (also used in the calculation of a state transition matrix) and covariance estimates in order to calculate the differential entropy of the system while estimates are propagated from time k to time $k+1$ [2, 35, 36]. Given the basic definition for differential entropy of a random variable $\vec{x}$ [10]

$$\mathcal{H}\left(\vec{x}\right) = -\int p\left(\vec{x}\right) \log p\left(\vec{x}\right) d\vec{x} \tag{3.61}$$

and taking $p\left(\vec{x}\right)$ to be Gaussian of the form in Eq. 3.1 and evaluating Eq. 3.61 the result (written in the nomenclature used in these studies) is

$$\mathcal{H}_{i,t}^{l} = \frac{1}{2}\log\left|2\pi e \hat{P}_{i,t}^{f,l}\right| \tag{3.62}$$

where $\hat{P}_{i,t}^{f,l}$ represents a forecast covariance estimate of component l in the GMM of object i evaluated at some propagation time t between the forecast time steps $k \rightarrow k+1$. Equation 3.62 serves as an easily computable metric which is used to determine the differential entropy of a nonlinear system, given some covariance estimate.

If the derivative of this system with respect to time is taken, using the matrix calculus property [35]

$$\frac{d}{dt}\left\{\left|\hat{P}_{i,t}^{f,l}\right|\right\} = \left|\hat{P}_{i,t}^{f,l}\right| \operatorname{tr}\left[\left(\hat{P}_{i,t}^{f,l}\right)^{-1} \frac{d}{dt}\left\{\hat{P}_{i,t}^{f,l}\right\}\right] \tag{3.63}$$

along with the linearized dynamics governing the covariance estimate [25]

$$\frac{d}{dt}\left\{\hat{P}_{i,t}^{f,l}\right\} = \left(\frac{\partial f}{\partial \vec{x}}\right)_{i,t} \hat{P}_{i,t}^{f,l} + \hat{P}_{i,t}^{f,l} \left(\frac{\partial f}{\partial \vec{x}}\right)_{i,t}^{T} \tag{3.64}$$

and the commonly known matrix trace operator property $\operatorname{tr}(A) = \operatorname{tr}\left(A^T\right)$, a differential equation for the linearized entropy is obtained

$$\dot{\mathcal{H}}_{i,t}^{l} = \operatorname{tr}\left[\left(\frac{\partial f}{\partial \vec{x}}\right)_{i,t}\right] \tag{3.65}$$

Therefore, two methods exist for calculating the differential entropy $\dot{\mathcal{H}}_{i,t}^{l}$. The first is a nonlinear case, in which Eq. 3.62 is evaluated at propagation time t using a covariance estimate $\hat{P}_{i,t}^{f,l}$ generated via nonlinear propagation (i.e. propagated using UKF methods described in Section 3.3). The second is a linear case in which Eq. 3.65 is evaluated using numerical integration techniques to find $\mathcal{H}_{i,t}^{l}$ using only the dynamics of the system and the current state estimate to evaluate $(\partial f/\partial \vec{x})_{i,t}$. As long as the linear case matches the nonlinear case, it is assumed that nonlinear effects are minute on the system, and a single Gaussian approximation for the PDF of component l is sufficient. Conversely, if a difference exists between the two cases, the indication is that nonlinear effects are increasing during propagation, and the component l should be split further into more GMM components to better approximate the now non-Gaussian PDF. Therefore, if the difference between the evaluation of $H_{i,t}^{l}$ using Eqs. 3.65 and 3.62 extends beyond a certain tolerance, it is assumed that nonlinear effects have influenced the propagation of $\hat{P}_{i,t}^{f,l}$, and further splitting on this component is done using methods described in Section 3.4.2. It should be noted that the evaluation of Eq. 3.62 is only of value if used within a UKF, or similar nonlinear propagation scheme. Should a linear propagation scheme be used (such as the EKF), there would be no difference between Eqs. 3.65 and 3.62, since both would reflect a linear case, and nonlinear effects would be undetectable.

3.5.5 The AEGIS Filter

By using a UKF filtering scheme, the AEGIS filter works in much the same way that a UKF does, except it applies the UKF approach to L components comprising a GMM which approximates a single Gaussian PDF reflected by an object's state (mean) and covariance estimates. The GMM, which is initially a single component Gaussian PDF described by $\hat{X}^*_{i,0}$ and $\hat{P}^*_{i,0}$, can be manipulated during the forecast step within a UKF propagation scheme to be split into multiple components as significant nonlinearity is detected in one or more GMM components. During this forecast step, the GMM maintains the constraints in Eq. 3.44, and will continue to add components to the GMM as needed until this step is completed.

3.5.5.1 Initialization

Starting with an initial distribution of sigma points calculated from[6]

$$\mathcal{X}_{i,0} = \hat{X}^*_{i,0}\,[1]_{1\times(2n)} + \sqrt{n}\left[0_{n\times 1}\ \hat{P}^{CH}_{i,0}\ -\hat{P}^{CH}_{i,0}\right] \tag{3.66}$$

These sigma points are propagated through the nonlinear system dynamics until sufficient effects of nonlinearity on the propagation are detected, or until the first time step comes to an end at $k = 1$.

3.5.5.2 Forecast Step

In a general propagation of sigma points from time step k to $k+1$, assuming that for some GMM component l of object i the sigma points at time step k are given by $\mathcal{X}^l_{i,k}$[7].

the sigma points of this component at time t in the propagation from time steps k to $k+1$ is

[6]Note that these $2n$ sigma points can create the same distribution mean and covariance as the $2n+1$ sigma point distribution illustrated in Eq. 3.32 and Section 3.3, as long as the constants κ, α, and β are chosen to reflect a Gaussian distribution where the weight associated with the additional sigma point calculated in Eq. 3.36 and 3.38 is zero (e.g. if $\kappa = 0$, $\alpha = 1$, and $\beta = 0$).

[7]For the initial Gaussian distribution, the GMM would only have one component, so this general set up for the AEGIS forecast step is valid even in the initial case when $L = 1$ and $k = 0$

$$\mathcal{X}_{i,t}^{l} = \mathcal{F}(\mathcal{X}_{i,k}^{l}) \tag{3.67}$$

where at time t in the numerical integration process, the state estimate is evaluated from

$$\hat{X}_{i,t}^{l} = \sum_{\gamma=1}^{2n} W_x^{\gamma} \mathcal{X}_{i,t}^{l,\gamma} \tag{3.68}$$

with weights W_x^{γ}

$$W_x^{\gamma} = \frac{1}{2n},\ \gamma = 1, \ldots, 2n \tag{3.69}$$

The Jacobian of the state dynamics can be evaluated at time step k by applying Eq. 2.6 with $\vec{x}_{i,t} = \hat{X}_{i,k}^{*}$ as an initial condition in the numerical integration of Eq. 3.65 to any integration time t, obtaining a linear approximation of $H_{i,t}^{l}$.

Likewise, at any time t the covariance estimate is determined from

$$\hat{P}_{i,t}^{l} = \sum_{\gamma=1}^{2n} W_p^{\gamma} \left[\mathcal{X}_{i,t}^{l,\gamma} - \hat{X}_{i,t}^{l}\right] \left[\mathcal{X}_{i,t}^{l,\gamma} - \hat{X}_{i,t}^{l}\right]^{T} \tag{3.70}$$

where $W_p^{\gamma} = W_x^{(\gamma)}\ \forall\gamma$. Next, this covariance estimate (recalling it was propagated nonlinearly) can be plugged into Eq. 3.62 to calculate the nonlinear differential entropy. If the difference between the nonlinear differential entropy and its linear approximation exceeds a user defined tolerance at time t, then the component l of the GMM describing object i must be split into $\tilde{L}$ components such that

$$\nu_{i,k}^{l} p^{g}\left(\vec{x}_{i,t}; \hat{X}_{i,t}^{l}, \hat{P}_{i,t}^{l}\right) \approx \sum_{\tilde{l}=1}^{\tilde{L}} \nu_{i,k}^{\tilde{l}} p^{g}\left(\vec{x}_{i,t}; \hat{X}_{i,t}^{\tilde{l}}, \hat{P}_{i,t}^{\tilde{l}}\right) \tag{3.71}$$

if $L > 1$, and

$$p^{g}\left(\vec{x}_{i,t}; \hat{X}_{i,t}^{f}, \hat{P}_{i,t}^{f}\right) \approx \sum_{\tilde{l}=1}^{\tilde{L}} \nu_{i,k}^{\tilde{l}} p^{g}\left(\vec{x}_{i,t}; \hat{X}_{i,t}^{\tilde{l}}, \hat{P}_{i,t}^{\tilde{l}}\right) \tag{3.72}$$

if $L = 1$. It should be observed that in Eqs. 3.71 - 3.72 the weights remain unchanged from their values at the time step k, implying that component weights remain constant during propagation, and are only altered if a splitting process occurs.

Splitting is achieved by applying the three component splitting library constants given in Table 3.1 to Eqs. 3.53 - 3.55 for the case of $L > 1$, and to Eqs. 3.56 - 3.58 if $L = 1$. For any l component that was split in such a way, sigma points for the $\tilde{L}$ components generated from this splitting are calculated from

$$\mathcal{X}_{i,t}^{\tilde{l}} = \hat{X}_{i,t}^{\tilde{l}} \left[1\right]_{1\times(2n)} + \sqrt{n}\left[0_{n\times 1}\ \hat{P}_{i,t}^{CH,\tilde{l}}\ -\hat{P}_{i,t}^{CH,\tilde{l}}\right],\ \forall \tilde{l} \tag{3.73}$$

where

$$\hat{P}_{i,t}^{CH,\tilde{l}}\left[\hat{P}_{i,t}^{CH,\tilde{l}}\right]^{T} = \hat{P}_{i,t}^{\tilde{l}} \tag{3.74}$$

After all sigma points are drawn for each new $\tilde{l}$ component of the $\bar{l}$ total original l components which were split, the number of components in the new GMM is reset such that $L \leftarrow L + \bar{l}\left(\tilde{L} - 1\right)$, and each component in the total GMM is once again referenced by the same nomenclature, l, such that $\sum_{l=1}^{L} l = L$. This process of detecting nonlinearity followed by splitting the GMM is continued until the propagation step is completed at the time step $k + 1$. Once this occurs, the forecast sigma points for each l component in the GMM are

$$\mathcal{X}_{i,k+1}^{f,l} = \mathcal{F}(\mathcal{X}_{i,t'}^{l}) \tag{3.75}$$

where t' is the last time between the time steps k and $k+1$ in which a splitting occurred. The forecast state and covariance estimates for the l^{th} component in the GMM are then evaluated by

$$\hat{X}_{i,k+1}^{f,l} = \sum_{\gamma=1}^{2n} W_x^{\gamma} \mathcal{X}_{i,k+1}^{f,l,\gamma} \tag{3.76}$$

and

$$\hat{P}^{f,l}_{i,k+1} = \sum_{\gamma=1}^{2n} W_p^{\gamma} \left[\mathcal{X}^{f,l,\gamma}_{i,k+1} - \hat{X}^{f,l}_{i,k+1}\right] \left[\mathcal{X}^{f,l,\gamma}_{i,k+1} - \hat{X}^{f,l}_{i,k+1}\right]^T + Q_{i,k} \tag{3.77}$$

3.5.5.3 Update Step

The update step in the AEGIS filter remains relatively unchanged from that in a standard UKF, except that the AEGIS update must not only provide updates for each l component in the GMM, but also update the weights associated with those components. Sorenson et. al. [31] originally came up with the method for updating these weights based upon applying Bayes rule to determine the a posterior GMM distribution conditional on the measurement data. For these studies, measurement data is used from all M' ($M' \leq M$) sensors tasked to observe object i (methods for determining objects tasked for observation in the set $\mathcal{O}^{\tau}_{k+1}$, as well as the sets of sensors tasked to observed them $\mathcal{S}^{\tau}_{i,k+1}$ are detailed in Chapter 4) at time step $k+1$ in the set $\mathcal{S}^{\tau}_{i,k+1} = \{s^{\tau}_1, \ldots, s^{\tau}_{M'}\}$ where elements in $\mathcal{S}^{\tau}_{i,k+1}$ represent a sensor index j. This measurement vector is given by Eq. 3.23.

Using the nomenclature of these studies, and constraining all distributions to be Gaussian, Sorenson et. al. determined the a posteriori distribution conditional on the total number of measurements $\vec{y}_{i,k+1}$ to be modeled as a GMM from

$$p^g\left(\vec{x}_{i,k+1}|\vec{y}_{i,k+1}\right) = p^g\left(\vec{x}_{i,k+1}; \hat{X}^*_{i,k+1}, \hat{P}^*_{i,k+1}\right) \approx \sum_{l=1}^{L} \nu^l_{i,k+1} p^g\left(\vec{x}_{i,k+1}; \hat{X}^{*,l}_{i,k+1}, \hat{P}^{*,l}_{i,k+1}\right) \tag{3.78}$$

where $\nu^l_{i,k+1}$ is the updated GMM component weight, given by

$$\nu^l_{i,k+1} = \frac{\nu^l_{i,k} p^g\left(\vec{y}_{i,k+1}; \hat{y}^l_{i,k+1}, P^{yy,l}_{i,k+1}\right)}{\sum_{l'=1}^{L} \nu^{l'}_{i,k} p^g\left(\vec{y}_{i,k+1}; \hat{y}^{l'}_{i,k+1}, P^{yy,l'}_{i,k+1}\right)} \tag{3.79}$$

Using UKF update techniques applied to each l component in the GMM representing object i, the measurement estimate $\hat{y}_{i,k+1}$ is calculated by inputting the sigma points for component l into the nonlinear measurement function such that

$$\mathcal{Y}_{i,k+1}^{l,\gamma} = \begin{bmatrix} h_\rho\left(\mathcal{X}_{i,k+1}^{f,l,\gamma}, \vec{s}_{s_1^\tau,k+1}\right) \\ \vdots \\ h_\rho\left(\mathcal{X}_{i,k+1}^{f,l,\gamma}, \vec{s}_{s_{M'}^\tau,k+1}\right) \\ h_\psi\left(\mathcal{X}_{i,k+1}^{f,l,\gamma}, \vec{s}_{s_1^\tau,k+1}\right) \\ \vdots \\ h_\psi\left(\mathcal{X}_{i,k+1}^{f,l,\gamma}, \vec{s}_{s_{M'}^\tau,k+1}\right) \end{bmatrix}, \ \gamma = 1, \ldots, 2n \tag{3.80}$$

The estimated measurement is calculated from this transformation of each sigma point through the measurement function by

$$\hat{y}_{i,k+1}^{l} = \sum_{\gamma=1}^{2n} W_x^\gamma \mathcal{Y}_{i,k+1}^{l,\gamma} \tag{3.81}$$

With the necessary calculations to perform the update step, the cross covariance is calculated by

$$P_{i,k+1}^{xy,l} = \sum_{\gamma=1}^{2n} W_p^\gamma \left[\mathcal{X}_{i,k+1}^{f,l,\gamma} - \hat{X}_{i,k+1}^{f,l}\right]\left[\mathcal{Y}_{i,k+1}^{l,\gamma} - \hat{y}_{i,k+1}^{l}\right]^T \tag{3.82}$$

while the innovation covariance is calculated by

$$P_{i,k+1}^{yy,l} = \sum_{\gamma=1}^{2n} W_p^\gamma \left[\mathcal{Y}_{i,k+1}^{l,\gamma} - \hat{y}_{i,k+1}^{l}\right]\left[\mathcal{Y}_{i,k+1}^{l,\gamma} - \hat{y}_{i,k+1}^{l}\right]^T \tag{3.83}$$

and the Kalman gain is obtained from

$$K_{i,k+1}^{l} = P_{i,k+1}^{xy,l}\left\{P_{i,k+1}^{yy,l} + R_{i,k+1}^{\tau}\right\}^{-1} \tag{3.84}$$

where $R_{i,k+1}^{\tau}$ is calculated using Eq. 3.29. Next, the GMM component weights are updated using Eq. 3.79, while the state and covariance estimates are updated from

$$\hat{X}_{i,k+1}^{*,l} = \hat{X}_{i,k+1}^{f,l} + K_{i,k+1}^{l}[\vec{y}_{i,k+1} - \hat{y}_{i,k+1}^{l}] \tag{3.85}$$

$$\hat{P}_{i,k+1}^{*,l} = \hat{P}_{i,k+1}^{f,l} - K_{i,k+1}^{l}\left[P_{i,k+1}^{yy,l} + R_{i,k+1}^{\tau}\right]\left(K_{i,k+1}^{l}\right)^T \tag{3.86}$$

3.5.6 Performance Discrepancies Between the UKF and AEGIS

Although the AEGIS filter reflects a recent development in the area of nonlinear estimation, DeMars et. al. have tested it within the nonlinear system of orbital dynamics, comparing its performance to the UKF [2, 35, 36]. Recently, they have also tested the AEGIS filter in regards to its effectiveness to calculate various information metrics (such as Renyi and Kullback-Leiber divergence) to be used in tracking problems similar to those found in SSA [60]. These preliminary studies have shown that the GMM approach to modeling and nonlinearly propagating a Gaussian PDF through orbital dynamics results in better uncertainty models as opposed to the single Gaussian PDF approach implemented by the UKF. The conjecture for these studies is that, much like comparing the EKF to the UKF, when comparing the UKF to the AEGIS filter, performance should be further increased using the AEGIS filter due primarily to the improved methods of uncertainty modeling.

In a similar fashion to Section 3.3, an illustration of these advantages in how the AEGIS filter better handles the calculation of forecast step uncertainty (reflected by the covariance estimate) than the UKF for nonlinear systems. Using the same example in Section 3.1.1, Figures 3.7 and 3.8 show how both the EKF and UKF propagate the initial mean given in Eq. 3.4 with an initial Gaussian covariance derived from the distribution generated from standard deviations in Eq. 3.5 as seen in Figure 3.1. Figure 3.9 shows how the AEGIS filter propagates these same initially (single component) Gaussian distributions, and how the resulting GMM PDF matches the uncertainty distribution better than the single Gaussian PDF's calculated using the EKF and UKF.

An algorithm which describes the step-by-step process of implementing the AEGIS filter for an object i (assuming measurement data exists for all M_s sensors at each time step) can be found in Appendix B.

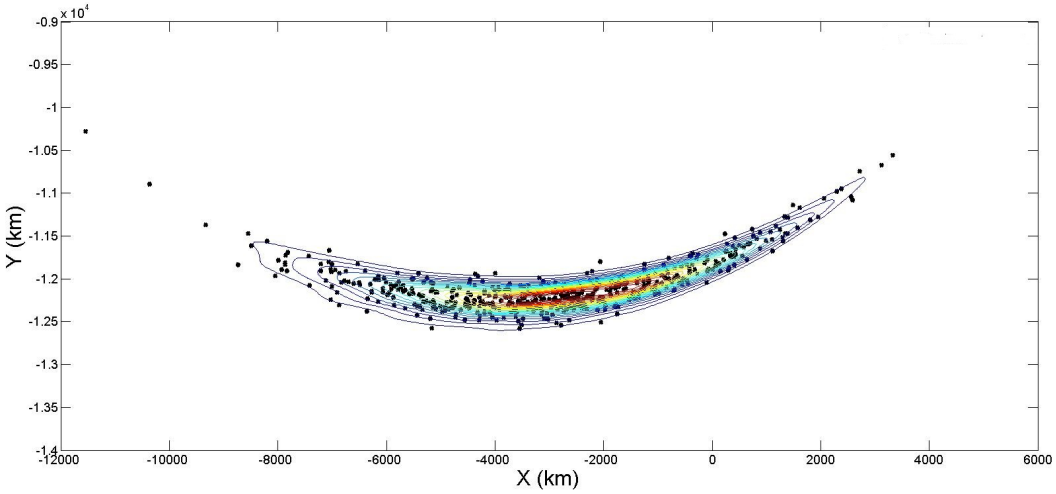

Figure 3.9. Propagation of initial Gaussian distribution with GMM PDF contour overlay calculated from propagating initial Gaussian PDF using AEGIS methods

Chapter 4

Sensor Tasking

Once the respective filters are applied to the multi-object satellite tracking problem, an issue arises regarding which objects to observe, and which to ignore should more than one be visible to a sensor at a certain time. This decision process is henceforth known as *sensor tasking*, and is implemented in coordination with the filtering process. In order to carry out these tasking decisions, some form of metric must be obtained, so that objects can be ranked (or prioritized) based on some performance criteria. This criteria takes the form of a *utility metric* which is gathered from the physical measurements taken on those objects, or the estimates of their states and covariances given by the filters. As follows are descriptions of three candidates for these utility metrics, and how they are obtained from the filter estimates and sensor measurements. Furthermore, each utility metric requires a separate decision process which determines the object-sensor pairs used for observation, given all of the possible object-sensor pairs at that time. Therefore, the tasking component of this simulation is segmented into determining the availible object-sensor pairs and assigning a utility metric to each, followed by a decision process which allocates sensor resources based on these metrics.

4.1 Tasking Problem Organization

Given forecast step object state and corresponding uncertainty estimates $\hat{X}^f_{i,k+1}$ and $\hat{P}^f_{i,k+1}$ described in Chapter 3, available object-sensor pairs (the sets $\mathcal{O}^a_{k+1}$

and $\mathcal{S}^a_{i,k+1}$ described in Chapter 2) are determined from Eqs. 2.17- 2.18, and two $N_s \times M_s$ matrices will be calculated.

The first, called the *visibility matrix*, $\underline{\mu}_{k+1}$ (containing elements $\mu_{(i,j),k+1}$) gives a numerical representation of the priority for measuring a particular object.

$$\underline{\mu}_{k+1} = \begin{bmatrix} \mu_{(1,1),k+1} & \cdots & \mu_{(1,M_s),k+1} \\ \vdots & \ddots & \vdots \\ \mu_{(N_s,1),k+1} & \cdots & \mu_{(N_s,M_s),k+1} \end{bmatrix} \tag{4.1}$$

Should a specific object i and sensor j pair be available for observation (i.e. $i \in \mathcal{O}^a_{k+1}$ and $j \in \mathcal{S}^a_{i,k+1}$), the element $\mu_{(i,j),k+1}$ will be determined by calculating a utility metric based on the method of tasking chosen. Should the object-sensor pair not be available for observation, the element $\mu_{(i,j),k+1}$ will be assigned a value that eliminates that object-sensor pair from being selected for tasking (for this studies, this will be $\mu_{(i,j),k+1} = 0$).

Assuming a constraint that only one object can be observed by a sensor at any particular time, the matrix $\underline{\mu}_{k+1}$ is then used in a decision making process which produces a second $N_s \times M_s$ matrix called the *decision matrix*, $\underline{\xi}_{k+1}$.

$$\underline{\xi}_{k+1} = \begin{bmatrix} \xi_{(1,1),k+1} & \cdots & \xi_{(1,M_s),k+1} \\ \vdots & \ddots & \vdots \\ \xi_{(N_s,1),k+1} & \cdots & \xi_{(N_s,M_s),k+1} \end{bmatrix} \tag{4.2}$$

The elements of this matrix $\xi_{(i,j),k}$ are binary where values of 0 indicate the object i is ignored (or simply not available) by sensor j, and 1 if satellite i is tasked to be observed by sensor j. While the calculation methods to obtain $\underline{\xi}_{k+1}$ will differ for each method of tasking, they will all be subject to a binary element and single object to one sensor constraints

$$\sum_{i=1}^{N_s} \xi_{(i,j),k+1} \leq 1, \ j = 1, \ldots, M_s \ \text{ and } \ \xi_{(i,j),k+1} \in [0,1]; \ \forall [i,j] \tag{4.3}$$

Methods of applying these constraints to calculate the decision matrix will revolve around either a continuation of previous work utilizing Fisher information

gain (FIG) as a tasking metric [21], or new methods utilizing largest Lyapunov exponent estimation (LLE) or Shannon information gain (SIG). For FIG tasking, a utility metric is calculated from the FIG for each available object-sensor pair, thus providing the elements for the visibility matrix. Next, A linear programming problem will be applied to the visibility matrix to determine a decision matrix which maximizes total information gained at a particular time step.

In SIG tasking, a small scale, yet simple dynamic programming problem is solved to select the object each sensor will observe at a given time step. In LLE tasking, elements of the visibility matrix for each available object-sensor pair will be determined from the LLE of each object. In this case, the decision process is greatly simplified so that each sensor views the object which has shown the greatest tendency towards divergence in its estimation error, given all the possible objects it can view at that time. Specific details of the calculation of the visibility and decision matrices using FIG, SIG, and LLE methods are found in Sections 4.2 - 4.4.

Once the elements of the decision matrix are calculated, indices of which objects are tasked to be observed $\mathcal{O}^{\tau}_{k+1}$ and the M' sensors tasked to observed them are determined from

$$\mathcal{O}^{\tau}_{k+1} = \left\{ i : \sum_{j=1}^{M_s} \xi_{(i,j),k+1} \geq 1 \right\} \tag{4.4}$$

and

$$\mathcal{S}^{\tau}_{i,k+1} = \left\{ j : \xi_{(i,j),k+1} = 1 \right\} \tag{4.5}$$

such that the number of indices in $\mathcal{S}^{\tau}_{i,k+1} \equiv M'$.

4.2 Fisher Information Gain

Fisher information takes the form of a matrix representing the amount of information present in an unbiased estimator. More specifically, it is used to approximate

the lower bound of the variance present in an unbiased estimator of some random variable, conditional upon some unknown parameter (in these studies, this random variable will be the states $\vec{x}$, and the unknown parameter the measurements $\vec{y}$). Through some derivation, it can be shown that the inverse of Fisher information provides a lower bound to the variance of the root-squared error of an unbiased estimator of $\vec{x}$, which is also known as the Cramer-Rao lower bound [15]. Since the Fisher information is the inverse to this lower bound, it follows that if the Fisher information is high, the variance in the root-squared error for the estimator of $\vec{x}$ will be low, hence the uncertainty in the estimator will also be low. Given the desire to reduce the overall uncertainty on the proposed multi-object tracking problem, Fisher information is a very useful metric in deciding which combinations of measurements from each possible object-sensor pairings will produce the largest increase in information (or greatest reduction in uncertainty).

In terms of probability theory, Fisher information is defined as the expected value (denoted as the $\mathbb{E}$ operator) of the variance of the score of a random variable, such that

$$\Sigma = \mathbb{E}\left[\left(\frac{\partial}{\partial \vec{x}} \log p\left(\vec{x}|\vec{y}\right)\right)^2\right] \tag{4.6}$$

where the Fisher information is given by Σ, and the probability density function of the random variable $\vec{x}$ conditional on $\vec{y}$ is given by $p\left(\vec{x}|\vec{y}\right)$.

An application of calculating this metric (in matrix form) to an estimator of a nonlinear system with additive (and constant) Gaussian process and measurement noise has been presented by Ristic [17] as the filtering information matrix, which when using the nomenclature of these studies is given by

$$\Sigma_{i,k+1} = \left(Q_{i,k+1} + \mathbb{E}\left[\tilde{F}_{i,k+1}\right] \hat{P}^*_{i,k} \mathbb{E}\left[\tilde{F}^T_{i,k+1}\right]\right)^{-1} + \mathbb{E}\left[\tilde{H}^T_{i,k+1}\right] R^{-1}_{k+1} \mathbb{E}\left[\tilde{H}_{i,k+1}\right] \tag{4.7}$$

where the matrices $\tilde{F}_{i,k+1}$ and $\tilde{H}_{i,k+1}$ are defined by the Jacobian's of the nonlinear state and measurement functions evaluated at the true state at time

$t = (k+1)\Delta t$. These can be evaluated with knowledge of all the M' sensors tasked to observe object i at time step $k+1$ in the set $\mathcal{S}^{\tau}_{i,k+1} = \{s^{\tau}_1, \ldots, s^{\tau}_{M'}\}$

$$\tilde{F}_{i,k+1} = \left[\nabla_{\vec{x}_{i,k+1}} f^T(\vec{x}_{i,k+1})\right]^T \tag{4.8}$$

$$\tilde{H}_{i,k+1} = \begin{bmatrix} \nabla_{\vec{x}_{i,k+1}} \begin{bmatrix} h_\rho\left(\vec{x}, \vec{s}_{s^{\tau}_1,k+1}\right) \\ \vdots \\ h_\rho\left(\vec{x}, \vec{s}_{s^{\tau}_{M'},k+1}\right) \\ h_\psi\left(\vec{x}, \vec{s}_{s^{\tau}_1,k+1}\right) \\ \vdots \\ h_\psi\left(\vec{x}, \vec{s}_{s^{\tau}_{M'},k+1}\right) \end{bmatrix}^T_{\vec{x}=\vec{x}_{i,k+1}} \end{bmatrix}^T \tag{4.9}$$

where $\nabla_{\vec{x}_{i,k+1}}$ referrs to applying the differential operator with respect to the current true state vector, $\vec{x}_{i,k+1}$. The $\left(Q_{i,k+1} + \mathbb{E}\left[\tilde{F}_{i,k+1}\right] \hat{P}^*_{i,k} \mathbb{E}\left[\tilde{F}^T_{i,k+1}\right]\right)^{-1}$ portion of Eq. 4.7 represents the information obtained from propagation of the system dynamics and covariance, while the information gained through observation $\vec{y}_{i,k+1}$ is

$$\tilde{\Omega}_{i,k+1} = \mathbb{E}\left[\tilde{H}^T_{i,k+1}\right] R^{-1}_{k+1} \mathbb{E}\left[\tilde{H}_{i,k+1}\right] \tag{4.10}$$

However, no estimator will give an exact analytical representation to $\tilde{H}_{i,k+1}$ since the true state $\vec{x}_{i,k+1}$ is never known, so it will need to be approximated. This approximation could be made by calculating the gradient of the truncated Taylor series centered about the forecast estimate, $\hat{X}^f_{i,k+1}$

$$\tilde{H}_{i,k+1} = \sum_{d=1}^{O} \frac{H^d_{i,k+1}\left(\vec{x}_{i,k+1} - \hat{X}^f_{i,k+1}\right)^{d-1}}{(d-1)!} \tag{4.11}$$

where the term O is the order of accuracy the particular filter is accurate to (making d the particular order of a term in the Taylor series expansion), and

$$H^d_{i,k+1} = \left[\begin{array}{c} \frac{\partial^d}{\partial \vec{x}} h_\rho \left(\vec{x}, \vec{s}_{s_1^\tau, k+1} \right) \\ \vdots \\ \frac{\partial^d}{\partial \vec{x}} h_\rho \left(\vec{x}, \vec{s}_{s_{M'}^\tau, k+1} \right) \\ \frac{\partial^d}{\partial \vec{x}} h_\psi \left(\vec{x}, \vec{s}_{s_1^\tau, k+1} \right) \\ \vdots \\ \frac{\partial^d}{\partial \vec{x}} h_\psi \left(\vec{x}, \vec{s}_{s_{M'}^\tau, k+1} \right) \end{array} \right]_{\vec{x}=\hat{X}^f_{i,k+1}} \tag{4.12}$$

Since the EKF linearizes the nonlinear measurement function, it is first order accurate and therefore for the EKF $\tilde{H}_{i,k+1}$ is derived from Equation 4.11

$$\tilde{H}^{EKF}_{i,k+1} = \left[\frac{\partial h\left(\vec{x}_{i,k+1}\right)}{\partial \vec{x}_{i,k+1}} \right]_{\vec{x}_{i,k+1}=\hat{X}^f_{i,k+1}} \tag{4.13}$$

However, since the UKF's estimated measurement is accurate to at least second order, and sometimes higher [27], it is more difficult to apply Eq. 4.11 to evaluate $\tilde{H}_{i,k+1}$ for the UKF. However, another approximation for $\tilde{H}_{i,k+1}$ is made, through investigating the information form of the Kalman Filter covariance update equation. Several publications [16, 14, 17, 21] have expressed this form when applied specifically to an EKF, but here it will be derived for the general case of the Kalman Filter algorithm (i.e. not making any assumptions concerning the accuracy of the approximations to the nonlinear measurement function).

To begin, it is necessary to define the inverse of the Kalman filter covariance update equation, which represents an approximation to Eq. 4.7. First, the innovation covariance $P^{yy}_{i,k+1}$ is defined in general terms as the expected value of the covariance of the estimated measurement with respect to the true measurement

$$P^{yy}_{i,k+1} = \mathbb{E}[(\vec{y}_{i,k+1} - \hat{y}_{i,k+1})(\vec{y}_{i,k+1} - \hat{y}_{i,k+1})^T] \tag{4.14}$$

which is rewritten as

$$P^{yy}_{i,k+1} = \begin{array}{c} \mathbb{E}[(\vec{y}_{i,k+1} - \hat{y}_{i,k+1})(\vec{x}_{i,k+1} - \hat{X}^f_{i,k+1})^T \left\{(\vec{x}_{i,k+1} - \hat{X}^f_{i,k+1})^T\right\}^{-1} \times \dots \\ \dots \times \left\{(\vec{x}_{i,k+1} - \hat{X}^f_{i,k+1})\right\}^{-1} (\vec{x}_{i,k+1} - \hat{X}^f_{i,k+1})(\vec{y}_{i,k+1} - \hat{y}_{i,k+1})^T] \end{array} \tag{4.15}$$

If the expectation operator is distributed, and the inverse exponent factored out, using the lemma $(AB)^{-1} = B^{-1}A^{-1}$, Eq. 4.15 is rearranged as

$$P^{yy}_{i,k+1} = \begin{array}{c} \mathbb{E}[(\vec{y}_{i,k+1} - \hat{y}_{i,k+1})(\vec{x}_{i,k+1} - \hat{X}^f_{i,k+1})^T] \times \dots \\ \dots \times \left\{\mathbb{E}[(\vec{x}_{i,k+1} - \hat{X}^f_{i,k+1})(\vec{x}_{i,k+1} - \hat{X}^f_{i,k+1})^T]\right\}^{-1} \times \dots \\ \dots \times \mathbb{E}[(\vec{x}_{i,k+1} - \hat{X}^f_{i,k+1})(\vec{y}_{i,k+1} - \hat{y}_{i,k+1})^T] \end{array} \tag{4.16}$$

Furthermore, the cross covariance is defined as the expected value of the covariance between the state forecast estimate with respect to the true state, and the measurement estimate with respect to the true measurement

$$P^{xy}_{i,k+1} = \mathbb{E}[(\vec{x}_{i,k+1} - \hat{X}^f_{i,k+1})(\vec{y}_{i,k+1} - \hat{y}_{i,k+1})^T] \tag{4.17}$$

Finally the forecast covariance is defined as the expected value of the covariance of the state forecast estimate with respect to the true state

$$\hat{P}^f_{i,k+1} = \mathbb{E}[(\vec{x}_{k+1} - \hat{X}^f_{k+1})(\vec{x}_{k+1} - \hat{X}^f_{k+1})^T] \tag{4.18}$$

When Equations 4.17 and 4.18 are applied to Equation 4.16, the result is

$$P^{yy}_{i,k+1} = \left[P^{xy}_{i,k+1}\right]^T \left\{\hat{P}^f_{i,k+1}\right\}^{-1} P^{xy}_{i,k+1} \tag{4.19}$$

This method for defining the innovation covariance in terms of the cross covariance and forecast covariance is valid for both the EKF and the UKF (and therefore the AEGIS) algorithms. By applying Eq. 4.19 to Eqs. 3.28 and 3.31, the covariance update equation is redefined as

$$\hat{P}^*_{i,k+1} = \hat{P}^f_{i,k+1} - \hat{P}^f_{i,k+1}\eta^T_{i,k+1}\left\{\eta_{i,k+1}\hat{P}^f_{i,k+1}\eta^T_{i,k+1} + R^\tau_{i,k+1}\right\}^{-1}\eta_{i,k+1}\hat{P}^f_{i,k+1} \tag{4.20}$$

where the matrix $\eta_{i,k+1}$ is defined by

$$\eta_{i,k+1} = \left[P^{xy}_{i,k+1}\right]^T \left\{\hat{P}^f_{i,k+1}\right\}^{-1} \tag{4.21}$$

If the matrix inversion lemma[1] is applied to Eq. 4.20, the information form of the Kalman Filter covariance update equation is obtained.

$$\left\{\hat{P}^*_{i,k+1}\right\}^{-1} = \left\{\hat{P}^f_{i,k+1}\right\}^{-1} + \eta^T_{i,k+1}\left\{R^\tau_{i,k+1}\right\}^{-1}\eta_{i,k+1} \tag{4.22}$$

Equation 4.22 is a Kalman filter-specific approximation for the filter information matrix, $\Sigma_{i,k+1}$, given in Eq. 4.7, and therefore an approximation to the Fisher information. Additionally, the term $\eta^T_{i,k+1}\left\{R^\tau_{i,k+1}\right\}^{-1}\eta_{i,k+1}$ is an approximation for the information gain, $\tilde{\Omega}_{i,k+1}$, given in Equation 4.10, making $\eta_{i,k+1}$ also an approximation for $\tilde{H}_{i,k+1}$ presented in Eq. 4.11.

Therefore, using $\eta_{i,k+1} \approx \tilde{H}_{k+1}$, the Kalman filter information gain matrix is described by

$$\hat{\Omega}_{i,k+1} = \eta^T_{i,k+1}\left\{R^\tau_{i,k+1}\right\}^{-1}\eta_{i,k+1} \tag{4.23}$$

This same information gain matrix has been referred to as the Fisher information gain (abbreviated as FIG in these studies) and has been suggested as a sensor tasking metric in the work of Tian, et. al. [14] and actually applied to a satellite tracking problem identical to the one used in these studies by Erwin, et. al. [21].

However, the measure of FIG in Eq. 4.23 represents the information gained by taking measurements from all sensors tasked to observe one object. For these studies, a calculation of FIG must be done of every possible object-sensor pair, and therefore Eq. 4.23 must be altered to take this into account. Therefore, to estimate the FIG for each object-sensor pair, Eq. 4.21 is reformulated to

$$\eta_{(i,j),k+1} = \left[P^{xy}_{(i,j),k+1}\right]^T \left\{\hat{P}^f_{i,k+1}\right\}^{-1} \tag{4.24}$$

such that the cross covariance $P^{xy}_{(i,j),k+1}$ is defined for a specific object i sensor

[1] $(A+BCD)^{-1} = A^{-1} - A^{-1}B(C^{-1} - DA^{-1}B)^{-1}DA^{-1}$

j pair. For an EKF this cross covariance is defined by

$$P^{xy}_{(i,j),k+1} = \hat{P}^{f}_{i,k+1} \left[H_{(i,j),k+1} \right]^{T} \tag{4.25}$$

with

$$H_{(i,j),k+1} = \left[\begin{array}{c} \frac{\partial}{\partial \vec{x}} h_{\rho} \left(\vec{x}, \vec{s}_{j,k+1} \right) \\ \frac{\partial}{\partial \vec{x}} h_{\psi} \left(\vec{x}, \vec{s}_{j,k+1} \right) \end{array} \right]_{\vec{x}=\hat{X}^{f}_{i,k+1}} \tag{4.26}$$

Should a UKF be implemented, the cross covariance would be calculated from

$$P^{xy}_{(i,j),k+1} = \sum_{\gamma=0}^{2n} W^{\gamma}_{p} \left[\mathcal{X}^{p,\gamma}_{i,k+1} - \hat{X}^{f}_{i,k+1} \right] \left[\mathcal{Y}^{\gamma}_{(i,j),k+1} - \hat{y}_{(i,j),k+1} \right]^{T} \tag{4.27}$$

with the sigma point measurements and estimated measurement calculated from[2]

$$\mathcal{Y}^{\gamma}_{(i,j),k+1} = \left[\begin{array}{c} h_{\rho} \left(\mathcal{X}^{p,\gamma}_{i,k+1}, \vec{s}_{j,k+1} \right) \\ h_{\psi} \left(\mathcal{X}^{p,\gamma}_{i,k+1}, \vec{s}_{j,k+1} \right) \end{array} \right], \ \gamma = 0, \ldots, 2n \tag{4.28}$$

and

$$\hat{y}_{(i,j),k+1} = \sum_{\gamma=0}^{2n} W^{\gamma}_{x} \mathcal{Y}^{\gamma}_{(i,j),k+1} \tag{4.29}$$

Whether an EKF or UKF is implemented, the resulting information form of the covariance update equation for object i based on measurements solely from the sensor j would be (recalling $R_{j,k+1}$ is defined in Eq. 2.16)

$$\left\{ \hat{P}^{*}_{(i,j),k+1} \right\}^{-1} = \left\{ \hat{P}^{f}_{i,k+1} \right\}^{-1} + \eta^{T}_{(i,j),k+1} \left[R_{j,k+1} \right]^{-1} \eta_{(i,j),k+1} \tag{4.30}$$

with the FIG for that object-sensor pair defined as

$$\hat{\Omega}_{(i,j),k+1} = \eta^{T}_{(i,j),k+1} \left[R_{j,k+1} \right]^{-1}_{k+1} \eta_{(i,j),k+1} \tag{4.31}$$

A convenience in defining the object-sensor pair specific FIG in Eq. 4.31 is

[2]Note that this is based on using $2n+1$ sigma points. Should a UKF with $2n$ sigma points be used (such as in the implementation of the AEGIS filter) γ values would simply change from $0 \ldots 2n$ to $0 \ldots 2n-1$ without change to Eqs. 4.28 - 4.29

that it can be used to obtain the total FIG for multiple sensors observing object i, as illustrated in the works of Erwin et. al.[21]. Considering a set of sensors $\mathcal{S}^{\tau}_{i,k+1} = \{s^{\tau}_1, \ldots, s^{\tau}_{M'}\}$ whose elements represent a sensor index j, Eq. 4.22 is modified to

$$\left\{\hat{P}^{*}_{i,k+1}\right\}^{-1} = \left\{\hat{P}^{f}_{i,k+1}\right\}^{-1} + \sum_{j'=1}^{M'} \eta^{T}_{\left(i,s^{\tau}_{j'}\right),k+1} R^{-1}_{s^{\tau}_{j'},k+1} \eta_{\left(i,s^{\tau}_{j'}\right),k+1} \tag{4.32}$$

Therefore, there exists a direct linear relationship between the FIG and the amount of information gained on a particular object, leading to a linear programming problem identical to that in the work of Erwin et. al.[21] which seeks to find the binary variables $\xi_{(i,j),k}$ which maximize

$$\sum_{i=1}^{N_s} \sum_{j=1}^{M_s} \mu_{(i,j),k+1}\, \xi_{(i,j),k+1} \tag{4.33}$$

subject to the constraints outlined in Eq. 4.3. This constrained maximization problem represents a linear programming problem and can be solved using a simplex method [61]. However, a problem arises in that FIG is computed as a matrix, and must be converted to a scalar for each object-sensor pair in order to solve Eq. 4.33. The two intuitive choices for this scalar metric would be to either take the trace or the determinant of the FIG in Eq. 4.31. In these studies, the trace will be used such that the elements of the visibility matrix are chosen as $\mu_{(i,j),k+1} = \phi_{(i,j),k+1}$ where

$$\phi_{(i,j),k+1} = \text{trace}\left(\left[\eta_{(i,j),k+1}\right]^{T} \left\{R_{j,k+1}\right\}^{-1} \eta_{(i,j),k+1}\right) \;\; \forall [i,j] \in [\mathcal{O}^{a}_{k+1}, \mathcal{S}^{a}_{i,k+1}] \tag{4.34}$$

In this case, the trace is used over the determinant, because the determinant of the FIG matrix in Eq. 4.31 will always be zero for an EKF, as illustrated by the simple example in Appendix A. The use the the trace is not without consequence, since it has no physical meaning to the problem because the units across diagonal elements of the FIG matrix are not equal (due to the position and velocity related components of the FIG matrix). This inconvenience could be alleviated

by either normalizing the diagonal components of the FIG matrix, or converting the matrix to canonical (unitless) dimensions, but this would in fact only change the scaling of elements in the matrix. The proper scaling between position and velocity information components would be a topic for further investigation, and more applicable to an investigation of the best practices for using an FIG-based tasking strategy applied to a satellite tracking problem. However, in the case of these studies, simply using the trace suffices to show trends in the coupling effect between the filters and the FIG matrix.

Thus, this tasking metric drives sensor allocation to provide the largest single-step increase in information to the collection of objects being tracked. Additionally, since this metric reflects an instantaneous benefit of taking a measurement on a certain object, without regard to future predictions of an object's uncertainty or knowledge of its dynamics, it can be viewed as a greedy or *myopic* form of tasking.

4.2.1 Calculating FIG for an EKF

By applying Eq. (4.25) to Eq. (4.24), the result is

$$\eta_{(i,j),k+1} = H_{(i,j),k+1} = \begin{bmatrix} \frac{\hat{x}^p_{i,k+1} - x^s_{j,k+1}}{\rho_{(i,j),k+1}} & \frac{\hat{y}^p_{i,k+1} - y^s_{j,k+1}}{\rho_{(i,j),k+1}} & 0 & 0 \\ -\frac{\hat{y}^p_{i,k+1} - y^s_{j,k+1}}{\rho^2_{(i,j),k+1}} & \frac{\hat{x}^p_{i,k+1} - x^s_{j,k+1}}{\rho^2_{(i,j),k+1}} & 0 & 0 \end{bmatrix} \tag{4.35}$$

which if applied to Eq. (4.34) results in a metric whose calculation depends only on the objects prediction step estimate location with respect to the sensor location, and the sensor noise covariance (which for these studies is constant for each sensor). This is an important observation, since it will become a factor in performance discrepancies between the EKF, and the UKF/AEGIS filters described in Chapter 6. Furthermore, this illustrates a direct consequence the EKF linearizations (in this case, the linearization of the nonlinear measurement function) have on the evaluation of the FIG metric.

4.2.2 Calculating FIG for a UKF

By applying Eq. (4.27) to Eq. (4.24), the result is

$$\eta_{(i,j),k+1} = \begin{array}{l} \left[\sum_{\gamma=0}^{2n} W_p^{\gamma} \left[\mathcal{X}_{i,k+1}^{f,\gamma} - \hat{X}_{i,k+1}^{f}\right] \left[\mathcal{Y}_{(i,j),k+1}^{\gamma} - \hat{y}_{(i,j),k+1}\right]^T\right]^T \times \ldots \\ \ldots \times \left\{\sum_{\gamma=0}^{2n} W_p^{\gamma} \left[\mathcal{X}_{i,k+1}^{f,\gamma} - \hat{X}_{i,k+1}^{f}\right] \left[\mathcal{X}_{i,k+1}^{f,\gamma} - \hat{X}_{i,k+1}^{f}\right]^T + Q_{i,k}\right\}^{-1} \end{array} \tag{4.36}$$

where the sigma point weights, W_p^{γ} are defined in Eqs. 3.36 and 3.38. If Eq. 4.36 is applied to Eq. (4.34) results in a metric that is a function of the distribution of sigma points as well as the constant sensor noise. In particular, both the distribution of the sigma points in the prediction step, and their transformation into measurement space will factor into the calculation of $\eta_{(i,j),k+1}$ and therefore $\phi_{(i,j),k+1}$. The evaluation of the FIG metric using a UKF differs significantly from that of an EKF, despite the fact that they represent the same quantity (that is a scalar evaluation of information gained from taking a measurement) for each filter.

4.2.3 Calculating FIG for an AEGIS filter

Unlike the EKF or UKF, the PDF describing the AEGIS filter is not a single Gaussian PDF, but rather a collection of weighted Gaussian PDF's resulting in a GMM PDF. The FIG could calculated by taking the inverse of Eq. 3.60, splitting it into components of information gained through propagation and information gained through measurement in the same manner as Eq. 4.22. However, this is not possible due to the fact that evaluation of Eq. 3.60 requires an updated state estimate for each l GMM component $\hat{X}_{i,k+1}^{*,l}$. In order to calculate this state estimate for each GMM component via Eq. 3.85, an actual measurement vector $\vec{y}_{i,k+1}$ would be impossible to obtain before any tasking decisions have been made. Therefore, any evaluation of FIG based on an object/sensor pair will have to be evaluated as an approximation to the true FIG.

This approximation is achieved by calculating a weighted average state and co-

variance estimates at the beginning of a time step via Eqs. 3.59 - 3.60, and applying a UKF forecast and update step to calculate $\eta_{(i,j),k+1}$ using a UKF strategy. This is the equivalent of calculating the UKF FIG of a single component GMM (which may approximate a multi-component GMM), and does not require measurements actually be taken as it would for the same calculation using a multi-component GMM.

Therefore, starting with the weighted average state and covariance estimates $\hat{X}_{i,k}^{*,gmm}$ and $\hat{P}_{i,k}^{*,gmm}$, $2n$ sigma points are drawn from

$$\mathcal{X}_{i,k}^{f,gmm} = \hat{X}_{i,k}^{f,gmm}\,[1]_{1\times(2n)} + \sqrt{n}\left[0_{n\times 1}\ \hat{P}_{i,k}^{CH,gmm}\ -\hat{P}_{i,k}^{CH,gmm}\right] \tag{4.37}$$

where

$$\hat{P}_{i,k}^{CH,gmm}\left[\hat{P}_{i,k}^{CH,gmm}\right]^T = \hat{P}_{i,k}^{*,gmm} \tag{4.38}$$

Sigma points are propagated in the same way as for the UKF described in Chapter 3 to obtain the forecast state $\left(\hat{X}_{i,k+1}^{f,gmm}\right)$ and covariance estimates $\left(\hat{P}_{i,k+1}^{f,gmm}\right)$. Next, the object-sensor cross covariance is calculated by

$$P_{(i,j),k+1}^{xy,gmm} = \sum_{\gamma=0}^{2n-1} W_p^{(\gamma)}\left[\mathcal{X}_{i,k+1}^{f,gmm,\gamma} - \hat{X}_{i,k+1}^{f,gmm}\right]\left[\mathcal{Y}_{(i,j),k+1}^{gmm,\gamma} - \hat{y}_{(i,j),k+1}^{gmm}\right]^T \tag{4.39}$$

where the sigma point measurements are

$$\mathcal{Y}_{(i,j),k+1}^{gmm,\gamma} = \left[\begin{array}{c} h_\rho\left(\mathcal{X}_{i,k+1}^{f,gmm,\gamma}, \vec{s}_{j,k+1}\right) \\ h_\psi\left(\mathcal{X}_{i,k+1}^{f,gmm,\gamma}, \vec{s}_{j,k+1}\right) \end{array}\right],\ \gamma = 0,\ldots,2n-1 \tag{4.40}$$

the estimated measurement is

$$\hat{y}_{(i,j),k+1}^{gmm} = \sum_{\gamma=0}^{2n-1} W_x^{\gamma}\mathcal{Y}_{(i,j),k+1}^{gmm,\gamma} \tag{4.41}$$

and the sigma point weights are defined in Eq. 3.69. Finally, $P_{(i,j),k+1}^{xy,gmm}$ and

$\hat{P}_{i,k+1}^{f,gmm}$ are applied to Eq. 4.24 to get $\eta_{(i,j),k+1}$, which is then used in Eq. 4.34 to obtain the approximate FIG for an AEGIS filter.

4.3 Shannon Information Gain

Defined as a scalar measure of information about the state of a system (applied here as a relative gain between a prior and posterior distribution), Shannon information is used in a similar manner to Fisher information as a utility metric for a sensor tasking problem. The primary differences between the two is that the SIG is normalized and represents a relative gain in information, while the FIG is an absolute gain. Utilizing the SIG as a tasking utility metric would therefore drive tasking decisions to observe objects which have the largest gain in information relative to where they were before observation while the FIG would task based on the maximum total gain of information. As follows is a brief description of how SIG is calculated, and how it is used within a single-step dynamic programming problem to determine tasking decisions.

To begin, the Shannon information originates from the definition of Shannon entropy, which measures chaos or disorder about the state of a system. Originally created to give a mathematical foundation to signal processing problems [62], entropy measures have been extended to applications in problems involving uncertainty measurements in statistical mechanics. For a differential system, the Shannon entropy is defined in Eq. 3.61. When assuming the PDF $p(\vec{x})$ is a Gaussian distribution $p^g\left(\vec{x}_{i,k}; \hat{X}_{i,k}^*, \hat{P}_{i,k}^*\right)$ for object i at time step k defined by Eq. 3.1, the result is

$$\mathcal{H}\left(\vec{x}_{i,k}\right) = -\int p^g\left(\vec{x}_{i,k}; \hat{X}_{i,k}^*, \hat{P}_{i,k}^*\right) \log p^g\left(\vec{x}_{i,k}; \hat{X}_{i,k}^*, \hat{P}_{i,k}^*\right) d\vec{x}_{i,k} \tag{4.42}$$

Shannon conjectures the similarities between this measure of entropy and the concept of information, in that the Shannon information was defined as the nega-

tive Shannon entropy[3]

$$I\left(\vec{x}_{i,k}\right) = \int p^g\left(\vec{x}_{i,k}; \hat{X}^*_{i,k}, \hat{P}^*_{i,k}\right) \log p^g\left(\vec{x}_{i,k}; \hat{X}^*_{i,k}, \hat{P}^*_{i,k}\right) d\vec{x}_{i,k} \tag{4.43}$$

By the definition that the expected value of a random variable x is $\mathbb{E}\left(x\right) = \int x f\left(x\right) dx$, Eq. 4.42 becomes

$$I\left(\vec{x}_{i,k}\right) = \mathbb{E}\left\{\log p^g\left(\vec{x}_{i,k}; \hat{X}^*_{i,k}, \hat{P}^*_{i,k}\right)\right\} \tag{4.44}$$

Evaluating the natural logarithm of $p^g\left(\vec{x}_{i,k}; \hat{X}^*_{i,k}, \hat{P}^*_{i,k}\right)$ yields

$$I\left(\vec{x}_{i,k}\right) = \mathbb{E}\left\{-\frac{1}{2}\left(\log\left|2\pi\hat{P}^*_{i,k}\right| + \text{trace}\left\{\left(\vec{x}_{i,k} - \hat{X}^*_{i,k}\right)\left(\vec{x}_{i,k} - \hat{X}^*_{i,k}\right)^T\left[\hat{P}^*_{i,k}\right]^{-1}\right\}\right)\right\} \tag{4.45}$$

If the expected value operator is distributed, the result is

$$I\left(\vec{x}_{i,k}\right) = -\frac{1}{2}\left(\log\left|2\pi\hat{P}^*_{i,k}\right| + \text{trace}\left\{\hat{P}^*_{i,k}\left[\hat{P}^*_{i,k}\right]^{-1}\right\}\right) \tag{4.46}$$

which leads to the final form of Shannon information for a Gaussian PDF

$$I\left(\vec{x}_{i,k}\right) = -\frac{1}{2}\log\left|2\pi e^n\hat{P}^*_{i,k}\right| \tag{4.47}$$

The Shannon information gain (SIG) is based on an evaluation of the information gained through observation from an a priori and a posteriori distribution as presented by several previous works. [13, 64, 65]. In this case, SIG is defined mathematically as the difference in Shannon information between the state at the forecast step (before observation) and the update step (after observation)

$$\Delta I_{i,k} = -\frac{1}{2}\log\left|2\pi e^n\hat{P}^*_{i,k+1}\right| - \left(-\frac{1}{2}\log\left|2\pi e^n\hat{P}^f_{i,k+1}\right|\right) \tag{4.48}$$

which is simplified to

[3]Several notions between entropy and information, including that made by Shannon can be found in the work of Skagerstam [63]

$$\Delta I_{i,k+1} = \frac{1}{2}\log\frac{\left|\hat{P}^{f}_{i,k+1}\right|}{\left|\hat{P}^{*}_{i,k+1}\right|} \tag{4.49}$$

When utilizing a Kalman filter while tracking N_s objects using M_s sensors, Eq. 4.49 for a particular object i observed by a sensor j at a time-step $k+1$ can be revised to become

$$\Delta I_{(i,j),k+1} = \frac{1}{2}\log\frac{\left|\hat{P}^{f}_{i,k+1}\right|}{\left|\hat{P}^{*}_{(i,j),k+1}\right|} \tag{4.50}$$

where calculating $\hat{P}^{*}_{(i,j),k+1}$ for a specific object-sensor pair is defined as

$$\hat{P}^{*}_{(i,j),k+1} = \hat{P}^{f}_{i,k+1} - K_{(i,j),k+1}\left[P^{yy}_{(i,j),k+1} + R_{j,k+1}\right]K^{T}_{(i,j),k+1} \tag{4.51}$$

In Eq. 4.51, the Kalman gain matrix is calculated from

$$K_{(i,j),k+1} = P^{xy}_{(i,j),k+1}\left\{P^{yy}_{(i,j),k+1} + R_{j,k+1}\right\}^{-1} \tag{4.52}$$

where the object-sensor specific innovation covariance is defined using the cross covariance (derivation in Section 4.2) from

$$P^{yy}_{(i,j),k+1} = \left[P^{xy}_{(i,j),k+1}\right]^{T}\left\{\hat{P}^{f}_{i,k+1}\right\}^{-1}P^{xy}_{(i,j),k+1} \tag{4.53}$$

with the object-sensor specific cross covariance defined for an EKF by Eqs. 4.25 - 4.26 and for a UKF by Eqs. 4.27 - 4.29. Therefore, for the purposes of using SIG as a tasking metric the elements of the visibility matrix are chosen as $\mu_{(i,j),k+1} = \Delta I_{(i,j),k+1}\ \forall\,[i,j] \in \left[\mathcal{O}^{a}_{k+1}, \mathcal{S}^{a}_{i,k+1}\right]$. However, a linear relationship between the total amount of SIG gained over all objects tasked for observation and the individual SIG for each of those tasked object-sensor pairs does not exist like it does for FIG, as illustrated by Eq. 4.32. In fact, the SIG calculated for an object-sensor pair will change each time the objects state changed (whether by propagation or by being observed by a sensor). That is, the addition of individual SIG's for an object and the M' sensors tasked to observe it is not equal to the total SIG of that object based on the observation of its M' sensors tasked to observe

it. Due to this, a solution to the decision matrix cannot be evaluated as a linear programming program as was the case with using FIG.

To formulate this decision making process, at each time-step a total possible SIG for each object (based on observations by all M' sensors tasked to observe it $\mathcal{S}^{\tau}_{i,k+1} = \{s^{\tau}_1, \ldots, s^{\tau}_{M'}\}$, and assuming only one observation is possible per sensor) is calculated from Eq. 4.49. This total SIG can also be broken down into a summation of iterative information gains from each sensor observing that object, due to the additive property of Shannon information [10]. Therefore, a total measure of the SIG can be redefined as

$$\Delta I_{i,k+1} = \Delta I_{\left(i,s^{\tau}_1\right),k+1} + \sum_{j'=1}^{M'-1} \Delta I_{\left(i,\left\{s^{\tau}_{j'+1},\ldots,s^{\tau}_1\right\}\right),k+1} \tag{4.54}$$

where $\Delta I_{\left(i,\left\{s^{\tau}_{j'+1},\ldots,s^{\tau}_1\right\}\right),k+1}$ represents the SIG between object i and sensor $j = s^{\tau}_{j'+1}$ based on observations from other sensors $j = \left\{s^{\tau}_{j'},\ldots,s^{\tau}_1\right\}$ tasked to observe that object. This individual SIG within the summation operator in Eq. 4.54 is calculated from[4]

$$\Delta I_{\left(i,\left\{s^{\tau}_{j'+1},\ldots,s^{\tau}_1\right\}\right),k+1} = \frac{1}{2}\log\frac{\left|\hat{P}^{*}_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_1\right\}\right),k+1}\right|}{\left|\hat{P}^{*}_{\left(i,\left\{s^{\tau}_{j'+1},\ldots,s^{\tau}_1\right\}\right),k+1}\right|},\ M' \neq 1 \tag{4.55}$$

where $\hat{P}^{*}_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_1\right\}\right),k+1}$ is the covariance estimate of object i based on the sensors $j = \left\{s^{\tau}_{j'},\ldots,s^{\tau}_1\right\}$ observing it[5]. To calculate this covariance estimate, a similar approach to the methods in Eqs. 3.31 and 4.51 are used, except that this new covariance estimate reflects an estimate based on only a portion of the sensors in $\mathcal{S}^{\tau}_{i,k+1}$ (unlike Eq. 3.31 which includes all sensors in $\mathcal{S}^{\tau}_{i,k+1}$, or Eq. 4.51 which includes only one sensor). This estimate is evaluated as

[4]Should $M' = 1$, $\Delta I_{i,k+1} \equiv \Delta I_{(i,j),k+1}$

[5]Likewise, $\hat{P}^{*}_{\left(i,\left\{s^{\tau}_{j'+1},\ldots,s^{\tau}_1\right\}\right),k+1}$ is the covariance estimate of object i based on the sensors $j = \left\{s^{\tau}_{j'+1},\ldots,s^{\tau}_1\right\}$ observing it

$$\hat{P}^{*}_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1} = \hat{P}^{f}_{i,k+1} - \ldots$$
$$\ldots - K_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1}\left[P^{yy}_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1} + R_{\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\},k+1}\right]K^{T}_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1} \tag{4.56}$$

In Eq. 4.56, the Kalman gain matrix is calculated from

$$K_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1} = P^{xy}_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1}\left\{P^{yy}_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1} + R_{\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\},k+1}\right\}^{-1} \tag{4.57}$$

where the innovation covariance is defined using the cross covariance

$$P^{yy}_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1} = \left[P^{xy}_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1}\right]^{T}\left\{\hat{P}^{f}_{i,k+1}\right\}^{-1}P^{xy}_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1} \tag{4.58}$$

and the sensor noise covariance matrix for the sensors $j = \left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}$ is

$$R_{\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\},k+1} = \begin{bmatrix} \left(v^{\rho}_{s^{\tau}_{1},k+1}\right)^{2} & 0 & \cdots & & \cdots & 0 \\ 0 & \ddots & & & & \vdots \\ \vdots & & \left(v^{\rho}_{s^{\tau}_{j'},k+1}\right)^{2} & & & \\ \vdots & & & \left(v^{\psi}_{s^{\tau}_{1},k+1}\right)^{2} & & \vdots \\ \vdots & & & & \ddots & 0 \\ 0 & \cdots & & \cdots & 0 & \left(v^{\psi}_{s^{\tau}_{j'},k+1}\right)^{2} \end{bmatrix} \tag{4.59}$$

Should an EKF be used, the cross covariance is calculated by

$$P^{xy}_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1} = \hat{P}^{f}_{i,k+1}\left[H_{\left(i,\left\{s^{\tau}_{j'},\ldots,s^{\tau}_{1}\right\}\right),k+1}\right]^{T} \tag{4.60}$$

with

$$H_{\left(i,\left\{s_{j'}^{\tau},\ldots,s_{1}^{\tau}\right\}\right),k+1}=\left[\begin{array}{c}\frac{\partial}{\partial\vec{x}}h_{\rho}\left(\vec{x},\vec{s}_{1,k+1}\right)\\ \vdots\\ \frac{\partial}{\partial\vec{x}}h_{\rho}\left(\vec{x},\vec{s}_{j',k+1}\right)\\ \frac{\partial}{\partial\vec{x}}h_{\psi}\left(\vec{x},\vec{s}_{1,k+1}\right)\\ \vdots\\ \frac{\partial}{\partial\vec{x}}h_{\psi}\left(\vec{x},\vec{s}_{j',k+1}\right)\end{array}\right]_{\vec{x}=\hat{X}_{i,k+1}^{f}} \tag{4.61}$$

If a UKF is implemented, this cross covariance is calculated by[6]

$$\begin{gathered}P_{\left(i,\left\{s_{j'}^{\tau},\ldots,s_{1}^{\tau}\right\}\right),k+1}^{xy}=\ldots\\ \ldots=\sum_{\gamma=0}^{2n}W_{p}^{\gamma}\left[\mathcal{X}_{i,k+1}^{f,\gamma}-\hat{X}_{i,k+1}^{f}\right]\left[\mathcal{Y}_{\left(i,\left\{s_{j'}^{\tau},\ldots,s_{1}^{\tau}\right\}\right),k+1}^{\gamma}-\hat{y}_{\left(i,\left\{s_{j'}^{\tau},\ldots,s_{1}^{\tau}\right\}\right),k+1}\right]^{T}\end{gathered} \tag{4.62}$$

where

$$\mathcal{Y}_{\left(i,\left\{s_{j'}^{\tau},\ldots,s_{1}^{\tau}\right\}\right),k+1}^{\gamma}=\left[\begin{array}{c}h_{\rho}\left(\mathcal{X}_{i,k+1}^{f,\gamma},\vec{s}_{s_{1}^{\tau},k+1}\right)\\ \vdots\\ h_{\rho}\left(\mathcal{X}_{i,k+1}^{f,\gamma},\vec{s}_{s_{j'}^{\tau},k+1}\right)\\ h_{\psi}\left(\mathcal{X}_{i,k+1}^{f,\gamma},\vec{s}_{s_{1}^{\tau},k+1}\right)\\ \vdots\\ h_{\psi}\left(\mathcal{X}_{i,k+1}^{f,\gamma},\vec{s}_{s_{j'}^{\tau},k+1}\right)\end{array}\right],\ \gamma=0,\ldots,2n \tag{4.63}$$

is used to calculate the estimated measurement

$$\hat{y}_{\left(i,\left\{s_{j'}^{\tau},\ldots,s_{1}^{\tau}\right\}\right),k+1}=\sum_{\gamma=0}^{2n}W_{x}^{\gamma}\mathcal{Y}_{\left(i,\left\{s_{j'}^{\tau},\ldots,s_{1}^{\tau}\right\}\right),k+1}^{\gamma} \tag{4.64}$$

An interesting property of Eq. 4.54 is that so long as two sets $\left\{s_{j'}^{\tau},\ldots,s_{1}^{\tau}\right\}$ utilize the exact same sensors, the particular order in which the updated posteriors (or the items in Eq. 4.54) occur is irrelevant. This means that for some given set of predetermined sensors, the order in which the gain in SIG is calculated to determine

[6]Note, this is based on a $2n+1$ sigma point distribution, and can easily be modified to incorporate a $2n$ sigma point case, such as in the AEGIS filter

the value for Eq. 4.54 will not alter the final value, and can be proven from[7]

$$\begin{gathered}\Delta I_{i,k+1} = \dots \\ \dots = \tfrac{1}{2}\log\frac{\left|\hat{P}^f_{i,k+1}\right|}{\left|\hat{P}^*_{\left(i,\left\{s^\tau_1\right\}\right),k+1}\right|} + \tfrac{1}{2}\log\frac{\left|\hat{P}^*_{\left(i,\left\{s^\tau_1\right\}\right),k+1}\right|}{\left|\hat{P}^*_{\left(i,\left\{s^\tau_2,s^\tau_1\right\}\right),k+1}\right|} + \dots \\ \dots + \tfrac{1}{2}\log\frac{\left|\hat{P}^*_{\left(i,\left\{s^\tau_{M'-1},\dots,s^\tau_1\right\}\right),k+1}\right|}{\left|\hat{P}^*_{i,k+1}\right|} = \tfrac{1}{2}\log\frac{\left|\hat{P}^f_{i,k+1}\right|}{\left|\hat{P}^*_{\left(i,\left\{s^\tau_1\right\}\right),k+1}\right|}\frac{\left|\hat{P}^*_{\left(i,\left\{s^\tau_1\right\}\right),k+1}\right|}{\left|\hat{P}^*_{\left(i,\left\{s^\tau_2,s^\tau_1\right\}\right),k+1}\right|} \times \dots \\ \dots \times \frac{\left|\hat{P}^*_{\left(i,\left\{s^\tau_{M'-1},\dots,s^\tau_1\right\}\right),k+1}\right|}{\left|\hat{P}^*_{i,k+1}\right|} = \tfrac{1}{2}\log\frac{\left|\hat{P}^f_{i,k+1}\right|}{\left|\hat{P}^*_{i,k+1}\right|}\end{gathered} \tag{4.65}$$

Given these properties, an optimal tasking decision process is chosen so that with an initial input reflecting the SIG present in each possible object sensor pair, the visibility matrix is calculated such that the elements $\mu_{(i,j),k+1} = \Delta I_{(i,j),k+1}$ $\forall\,[i,j] \in \left[\mathcal{O}^a_{k+1}, \mathcal{S}^a_{i,k+1}\right]$. This decision process selects the binary variables $\xi^j_{i,k+1}$ that maximizes the total instantaneous SIG across all objects observed

$$\Delta I^{TOT}_{k+1} = \sum_{i=1}^{N_s} \Delta I_{i,k+1} \tag{4.66}$$

while maintaining the constraints presented in Eq. 4.3.

The solution to Eq. 4.66 represents the solution to a dynamic programming problem, due to the fact that after the selection of a sensor to observe a particular object, the SIG for that object based on the remaining sensors (should more than one sensor be available to this object) will need to be recalculated due to the change in the PDF resulting from that observation. This fact, coupled with the memoryless properties of the individual SIG's (i.e. the calculation of the SIG only depends on the current PDF in the sequence, and not previous ones) define the calculation of total SIG in Eq. 4.54 to be a Markov chain. While many dynamic programming problems require difficult and computationally expensive solutions

[7] Understanding that $\hat{P}^*_{\left(i,\left\{s^\tau_{M'},\dots,s^\tau_1\right\}\right),k+1} \equiv \hat{P}^*_{i,k+1}$

processes, the fact that the total SIG calculated for an object is a Markov chain, it can be shown that the solution to this optimization problem is very straightforward.

To illustrate this optimal process, a a brief example is illustrated. First, assume that there exist $M = 1$ sensors to observe $N = 4$ objects, $A \rightarrow D$, such that the visibility matrix for this case would be

$$\underline{\mu}_{k+1} = \left[\begin{array}{cccc} \Delta I_{(A,1),k+1} & \Delta I_{(B,1),k+1} & \Delta I_{(C,1),k+1} & \Delta I_{(D,1),k+1} \end{array} \right] \tag{4.67}$$

where for this example, $\Delta I_{(A,1),k+1} > \Delta I_{(B,1),k+1} > \Delta I_{(C,1),k+1} > \Delta I_{(D,1),k+1}$. Obviously, given that only one decision exists (due to the one object per sensor constraint) the optimal decision is that $\mathcal{S}^{\tau}_{A,k+1} = \{1\}$ and $\mathcal{S}^{\tau}_{B,k+1} \rightarrow \mathcal{S}^{\tau}_{D,k+1} = \{0\}$.

Adding an additional degree of complexity, assume another sensor is added to the previous example, such that now $M = 2$. Assume that the SIG for each object-sensor pair in Eq. 4.67 remains unchanged, but now an additional row for the Shannon information based on observations by the second sensor is added, such that the visibility matrix is now

$$\underline{\mu}_{k+1} = \left[\begin{array}{cccc} \Delta I_{(A,1),k+1} & \Delta I_{(B,1),k+1} & \Delta I_{(C,1),k+1} & \Delta I_{(D,1),k+1} \\ \Delta I_{(A,2),k+1} & \Delta I_{(B,2),k+1} & \Delta I_{(C,2),k+1} & \Delta I_{(D,2),k+1} \end{array} \right] \tag{4.68}$$

Due to the increased dimensionality of this new visibility matrix, the complexity will increase in determining the correct sensor decisions to maximize Eq. 4.66. Assume that $\Delta I_{(A,1),k+1} > \Delta I_{(A,2),k+1}$, and focus on object A. From Eqs. 4.54 and 4.65, it follows that the maximum possible SIG obtainable for object A is

$$\Delta I_{A,k+1} = \Delta I_{(A,1),k+1} + \Delta I_{(A,\{2,1\}),k+1} = \Delta I_{(A,2),k+1} + \Delta I_{(A,\{1,2\}),k+1} \tag{4.69}$$

From Eq. 4.69 and the given property that $\Delta I_{(A,1),k+1} > \Delta I_{(A,2),k+1}$, then

$$\Delta I_{(A,\{1,2\}),k+1} > \Delta I_{(A,\{2,1\}),k+1} \tag{4.70}$$

Additionally, the work of Aughenbaugh et. al.[13] illustrates that the individual

SIG's in Eq. 4.55 are equal to the Kullback-Lieber divergence between the prior and posterior distributions. The Kullback-Leiber divergence, which represents the mutual information between the state before and after a particular observation occurs, was derived by Hershey et. al. [66] for Gaussian posterior and prior distributions as (modifying the original equation to incorporate a forecast step prior and update step posterior defined in the nomenclature of these studies)

$$\begin{aligned}\mathcal{D}_{KL}\left(p^g\left(\vec{x}_{i,k+1};\hat{X}^f_{i,k+1},\hat{P}^f_{i,k+1}\right),p^g\left(\vec{x}_{i,k+1};\hat{X}^*_{i,k+1},\hat{P}^*_{i,k+1}\right)\right)=\dots\\ \dots=\tfrac{1}{2}\log\frac{\left|\hat{P}^f_{i,k+1}\right|}{\left|\hat{P}^*_{i,k+1}\right|}+\tfrac{1}{2}\text{trace}\left[\left\{\hat{P}^f_{i,k+1}\right\}^{-1}\hat{P}^*_{i,k+1}\right]+\dots\\ \dots+\tfrac{1}{2}\left[\hat{X}^*_{i,k+1}-\hat{X}^f_{i,k+1}\right]^T\left\{\hat{P}^f_{i,k+1}\right\}^{-1}\left[\hat{X}^*_{i,k+1}-\hat{X}^f_{i,k+1}\right]-\tfrac{n}{2}\end{aligned} \tag{4.71}$$

Aughenbaugh applied this calculation of the Kullback-Leibler divergence to a Kalman filter algorithm, where the posterior covariance is independent of the measurement $\vec{y}_{i,k+1}$, to find that in this particular case

$$\mathcal{D}_{KL}\left(p^g\left(\vec{x}_{i,k+1};\hat{X}^f_{i,k+1},\hat{P}^f_{i,k+1}\right),p^g\left(\vec{x}_{i,k+1};\hat{X}^*_{i,k+1},\hat{P}^*_{i,k+1}\right)\right)=\frac{1}{2}\log\frac{\left|\hat{P}^f_{i,k+1}\right|}{\left|\hat{P}^*_{i,k+1}\right|} \tag{4.72}$$

Therefore, the SIG calculated in these studies is in fact equivalent to the mutual information between a prior and posterior distribution, $\Delta I_{i,k+1}\equiv\mathcal{I}\left(\vec{x}_{i,k+1};\vec{y}_{i,k+1}\right)$[8] [64]. This fact, coupled with the definition of the total SIG calculation in Eq. 4.54 being a Markov chain can be used within the data processing inequality to show that $\mathcal{I}\left(\vec{x}_{i,k+1};\vec{y}_{\left(i,\left\{s^\tau_{j'},\dots,s^\tau_1\right\}\right),k+1}\right)\leq\mathcal{I}\left(\vec{x}_{i,k+1};\vec{y}_{\left(i,\left\{s^\tau_1\right\}\right),k+1}\right)\ \forall j'>1$ [15].

Using the nomenclature of the current example, this implies that

$$\begin{aligned}\Delta I_{(A,1),k+1}&>\Delta I_{(A,\{1,2\}),k+1}\\ \Delta I_{(A,2),k+1}&>\Delta I_{(A,\{2,1\}),k+1}\end{aligned} \tag{4.73}$$

This inequality could easily be expanded to more than 2 sensors with the same

[8] $\mathcal{I}\left(\vec{x}_{i,k+1};\cdot\right)$ could represent a mutual information, or a conditional mutual information based on how many sensors have already been tasked to observe object i in a particular instant of time.

results, translating to a general form of

$$\Delta I_{(i,j),k+1} \geq \Delta I_{\left(i,\left\{s^{\tau}_{M'},\ldots,s^{\tau}_{1}\right\}\right),k+1},\ \text{for } j = s^{\tau}_{M'} \text{ and } 2 \leq M' \tag{4.74}$$

which would be valid for any set of sensors $\mathcal{S}^{\tau}_{i,k+1} \subseteq \mathcal{S}^{a}_{i,k+1}$ where $j = s^{\tau}_{M'}$. In a general description, Eq. 4.74 states that the SIG between an object i and sensor j is always greater than or equal to the SIG between object i and sensor j conditional on observations from any additional sensors. However, this inequality can be expanded due to the fact that $\Delta I_{(A,1),k+1} > \Delta I_{(A,2),k+1}$, implying that if

$$\Delta I_{(i,j_{max}),k+1} > \Delta I_{(i,j),k+1}\ \forall\ j \neq j_{max} \tag{4.75}$$

then

$$\Delta I_{(i,j_{max}),k+1} > \Delta I_{\left(i,\left\{s^{\tau}_{M'},\ldots,s^{\tau}_{1}\right\}\right),k+1},\ 2 \leq M' \tag{4.76}$$

and is valid for all $\mathcal{S}^{\tau}_{i,k+1} \subseteq \mathcal{S}^{a}_{i,k+1}$. In Eq. 4.76, j_{max} represents the sensor j associated with the highest single-sensor SIG for some object i, such that

$$j_{max} = \max_{j} \Delta I_{(i,j),k+1} \tag{4.77}$$

The inequality presented in Eqs. 4.75 - 4.76 can also be expanded to the current sensor tasking example, when taking into account the SIG present in each object-sensor pair in $\underline{\mu}_{k+1}$. That is, using the same analysis methods applied previously, it can be shown that if

$$\Delta I_{(i_{max},j_{max}),k+1} > \Delta I_{(i,j),k+1}\ \forall\ [i,j] \neq [i_{max},j_{max}] \tag{4.78}$$

then

$$\Delta I_{(i_{max},j_{max}),k+1} > \Delta I_{\left(i,\left\{s^{\tau}_{M'},\ldots,s^{\tau}_{1}\right\}\right),k+1},\ 2 \leq M' \tag{4.79}$$

for all $\mathcal{S}^{\tau}_{i,k+1} \subseteq \mathcal{S}^{a}_{i,k+1}$ and all $\mathcal{O}^{\tau}_{k+1} \in \mathcal{O}^{a}_{k+1}$. In Eq. 4.79, $[i_{max}, j_{max}]$ represents the object-sensor pair associated with the highest single-sensor SIG, such that i_{max} and j_{max} are calculated from

$$[i_{max}, j_{max}] = \max_{i,j} \Delta I_{(i,j),k+1} \tag{4.80}$$

The inequality presented in Eqs. 4.78 - 4.79 is fundamental in the solution to the dynamic programming problem that finds the set of objects $\mathcal{O}^{\tau}_{k+1}$ and sensors observing those objects $\mathcal{S}^{\tau}_{i,k+1}$ (and therefore finding the elements $\xi_{(i,j),k+1}$) which maximize Eq. 4.66. This is because Eqs. 4.78 - 4.79 state that as long as at each tasking decision the object-sensor pair corresponding to the highest value of SIG available for all feasible object-sensor pairs is selected, then no other sequence of tasking decisions will produce a greater value of Eq. 4.66.

Therefore, in the current example, if $\Delta I_{(A,1),k+1} = \mu_{(A,1),k+1}$ is the entry in $\underline{\mu}_{k+1}$ with the greatest magnitude, then first tasking decision should be that sensor 1 observes object A, making $\xi_{(A,1),k+1} = 1$. Since after this point the estimation state (in this case the covariance) of object A will be altered, the next step would be to recalculate $\mu_{(A,2),k+1} = \Delta I_{(A,\{2,1\}),k+1}$ so it reflects the current updated SIG present in the object A - sensor 2 pair. Furthermore, since sensor 1 would no longer be available to make any observations, setting $\mu_{(A,1),k+1} \rightarrow \mu_{(D,1),k+1} = 0$ would ensure that no other decisions could be made to select sensor 1 for another observation. Implementing these changes, for the next decision sequence the visibility matrix would be changed to

$$\underline{\mu}_{k+1} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ \Delta I_{(A,\{2,1\}),k+1} & \Delta I_{(B,2),k+1} & \Delta I_{(C,2),k+1} & \Delta I_{(D,2),k+1} \end{bmatrix} \tag{4.81}$$

which has the easy decision of selecting the object with the highest SIG (the same as in the first example), whichever object that may be. Assuming that $\Delta I_{(B,2),k+1} > \Delta I_{(C,2),k+1} > \Delta I_{(D,2),k+1} > \Delta I_{(A,\{2,1\}),k+1}$, the next decision would be to have sensor 2 observe object B, which would result in a decision matrix of

$$\underline{\xi}_{k+1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{4.82}$$

In this example, the set of objects to be observed would be $\mathcal{O}^{\tau}_{k+1} = \{A, B\}$ and

the sets of sensors observing each object would be $\mathcal{S}^{\tau}_{A,k+1} = \{1\}$, $\mathcal{S}^{\tau}_{B,k+1} = \{2\}$, and $\mathcal{S}^{\tau}_{C,k+1} \rightarrow \mathcal{S}^{\tau}_{D,k+1} = \{0\}$. Therefore, the optimal decision process illustrated in the above examples can be expended to a case with M_s sensors and N_s objects, to be embedded within a Kalman filter algorithm after the forecast step and before the update step. This decision process would simply initiate $\underline{\xi}_{k+1} = \mathbf{0}_{N_s \times M_s}$ and given $\underline{\mu}_{k+1}$ would

- Identify the object-sensor pair $[i_{max}, j_{max}]$ with largest $\mu_{(i,j),k+1}$
- Set $\xi_{(i_{max},j_{max}),k+1} = 1$
- Set $\mu_{(i,j_{max}),k+1} = 0 \ \forall i$
- If $\underline{\mu}_{k+1} \neq \mathbf{0}_{N_s \times M_s}$ recalculate $\Delta I_{(i_{max},\{j,j_{max},\ldots\}),k+1} \ \forall j \neq j_{max}$ and repeat above steps, otherwise the decision process is complete

4.3.1 Calculating SIG for an EKF

The process for calculating the SIG is very straightforward for an EKF. Given a forecast estimate $\hat{P}^{f}_{i,k+1}$, an update covariance estimate for object i and sensor j is calculated using Eq. 4.51. These estimates are used in Eq. 4.50 to calculate the initial SIG for the object i sensor j pair. Should object i be tasked for observation by sensor j and its SIG need to be recalculated as part of the decision process, the new SIG for object i and a new sensor j' (where the original sensor to task object i is represented by $j = s^{\tau}_1$) is calculated from

$$\Delta I_{\left(i,\left\{j',s^{\tau}_1\right\}\right),k+1} = \frac{1}{2} \log \frac{\left|\hat{P}^{*}_{\left(i,s^{\tau}_1\right),k+1}\right|}{\left|\hat{P}^{*}_{\left(i,\left\{j',s^{\tau}_1\right\}\right),k+1}\right|} \tag{4.83}$$

where $\hat{P}^{*}_{\left(i,\left\{j',s^{\tau}_1\right\}\right),k+1}$ is calculated using Eq. 4.56.

4.3.2 Calculating SIG for a UKF

The process for calculating the SIG for a UKF is very similar to that of an EKF. Given a forecast estimate $\hat{P}^{f}_{i,k+1}$, an update covariance estimate for object i and

sensor j is calculated using Eq. 4.51. These estimates are used in Eq. 4.50 to calculate the initial SIG for the object i sensor j pair. Should object i be tasked for observation by sensor j and its SIG need to be recalculated as part of the decision process, the new SIG for object i and a new sensor j' (where the original sensor to task object i is represented by $j = s_1^\tau$) is calculated from Eq. 4.83 where $\hat{P}^*_{\left(i,\left\{j',s_1^\tau\right\}\right),k+1}$ is calculated using Eq. 4.56.

4.3.3 Calculating SIG for the AEGIS filter

The process for calculating the SIG for an AEGIS filter is similar to the FIG, in that its an approximation, since no closed loop solution exists. Given that there is no way to take a measurement before tasking decisions have been made, any tasking utility metric cannot be based upon knowledge of a measurement, and therefore calculating SIG cannot be done for a GMM using the methods presented in this section. However, like the FIG, if during the tasking process, the multi-component GMM is reduced to a single Gaussian PDF (with one state and covariance estimate), a traditional UKF calculation method is implemented to gain an approximation for the SIG.

Given state $\left(\hat{X}^{f,gmm}_{i,k}\right)$ and covariance $\left(\hat{P}^{f,gmm}_{i,k}\right)$ estimates calculated using Eqs. 3.59 - 3.60, sigma points are distributed using Eq. 4.37 to obtain $\mathcal{X}^{f,gmm}_{i,k}$. These sigma points are propagated through the nonlinear dynamics, and forecast estimates are obtained using Eqs. 3.35 and 3.37. Additionally, Eqs. 4.39 - 4.41 are used to gain the cross covariance, measurement sigma points, and estimated measurement, while the innovation covariance is calculated from Eq. 4.53. These are applied to Eqs. 4.51 - 4.52 to obtain the Kalman gain and covariance updates, which are inputted into Eq. 4.50 (using the approximation $\hat{P}^{f,gmm}_{i,k+1} \approx \hat{P}^{f}_{i,k+1}$) to gain the SIG for object i and sensor j.

Should object i be tasked for observation by sensor j and its SIG need to be recalculated as part of the decision process, the new SIG for object i and a new sensor j' (where the original sensor to task object i is represented by $j = s_1^\tau$) is calculated from Eq. 4.83 where $\hat{P}^*_{\left(i,\left\{j',s_1^\tau\right\}\right),k+1}$ is calculated using Eq. 4.56.

The application of Eq. 4.56 for an AEGIS filter uses the same equations as for application with a UKF, except the the number of sigma points are changed from $2n+1$ to $2n$, and sigma point weights are calculated using Eq. 3.69.

4.4 Lyapunov Exponent Metrics

While metrics involving information theory give insights concerning the reduction of uncertainty within a particular measurement of an object, their single-step myopic application in these studies may result in little to no observations of objects having low values of FIG or SIG with respect other observable objects. This could lead to a particular object not being observed for long periods of time, possibly leading to divergence in its uncertainty to the extent of the object becoming unable to monitor. Given that divergence is inherently tied into stability for dynamic systems, it follows that some method of assessing the stability of the estimation error and/or uncertainty may also provide a valuable tasking metric to examine. In modern nonlinear control theory, as well as data analysis, the concept of assessing the stability of a dynamic system has been thoroughly studied through methods of Lyapunov stability analysis, including Lyapunovs direct method [67]. For time-series of data, these methods include an approach called *largest Lyapunov exponent* estimation [10] (LLE), commonly performed in applications containing collections of stochastic data.

The motivation to consider a tasking metric based on Lyapunov exponents is due to the non-predictive nature of the myopic single time step FIG and SIG-based metrics. Lyapunov exponents, while difficult to calculate in the absence of substantial data, could prove to be a suitable metric for this purpose. Lyapunov exponents provide insight into the stability a dynamical system by calculating a series of exponents known as a *Lyapunov spectrum* which reflects the extent of divergence between the trajectories of two or more neighboring points in state space. More specifically, they are a measurement of exponential divergence of two possible points propagated through state-space dynamics over an infinite time in which a Lyapunov exponent exists for every state dimension in the system. Within the Lyapunov spectrum, the *largest Lyapunov exponent* has the greatest magnitude,

and in many cases when system dynamics cannot be modeled analytically, the calculation of the largest exponent can serve as an approximation to the spectrum.[10]

Mathematically, for two data points at some time t, $x_{1,t}$ and $x_{2,t}$, having a distance $\|x_{1,t} - x_{2,t}\| = \delta_t << 1$, which when measured after some time Δt will have a distance $\|x_{1,t+\Delta t} - x_{2,t+\Delta t}\| = \delta_{t+\Delta t}$, a Lyapunov exponent λ can be approximated with

$$\delta_{t+\Delta t} \cong \delta_t e^{\lambda \Delta t} \tag{4.84}$$

Therefore, for a value of $\lambda \leq 0$ the points are converging, if $\lambda > 0$ the points are diverging (to varying extents, with $\lambda > 1$ representing exponentially fast divergence), and if $\lambda = \infty$ the data is pure chaos (noise). Additionally, the sum of the Lyapunov exponents has the same property, in that if their summation $\sum_{m=1}^{n} \lambda^{(m)}$ is greater than zero, the system is unstable.

Various methods exist in determining these Lyapunov exponents, including estimating the entire spectrum based on a deterministic system model, or estimating only the largest exponent based on a time series of data [37, 68, 69, 70]. Should system dynamics be modeled deterministically, for example, via a state transition matrix like that in Eq. 2.4, the spectrum of exponents is calculated from time-steps $k = 0 \rightarrow k = K'$ by

$$\lambda_i^{(m)} = \lim_{K' \rightarrow \infty} \frac{1}{2K'} ln \left(\left| \mathcal{L}_{i,K'}^{(m)} \right| \right) \tag{4.85}$$

where

$$\mathcal{L}_{i,K'}^{(m)} \vec{u}_i^{(m)} = \left(\prod_{k=0}^{K'} \Phi_{i,k|k+1} \right)^T \left(\prod_{k=0}^{K'} \Phi_{i,k|k+1} \right) \vec{u}_i^{(m)} \tag{4.86}$$

represents a system of linear equations in the coefficients of $\vec{u}_i^{(m)}$, making $\mathcal{L}_{i,K'}^{(m)}$ and eigenvalue. Furthermore, the sum of these exponents is calculated to give an indication of the instability of each of the orbits of the various simulated objects. This calculation is done, starting with the initial estimates of each object and propagating Eqs. 2.2 and 2.5 forward to some terminal time step K. This method

of calculating the entire spectrum could be applied to the satellite equations of motion to give an indication of which objects would be subject to the greatest state space instability. However, while the stability of a satellites equations of motion may give an indication of which satellites should be observed the most, it does not incorporate any relation to estimation, and is therefore not considered for these studies.

However, if a deterministic model does not exist for the system (such as in a stochastic time series of data produced by a filter) Lyapunov exponents can be estimated by examining exponential divergence in a time series of data. As time between updates (or observations) of an object increase, its estimation error may also increase, if not diverge drastically should this time be large. The extent of this divergence will be related to how the actual state and the estimated state separate over time, as reflected by the propagation of the covariance estimate. Using both an EKF and UKF, this propagation of covariance is based upon the modeling of the object dynamics, such that if the dynamics dictates that two close initial states will diverge when propagated through the equations of motion, the hyper volume of the covariance estimate should diverge in kind. Since this hyper volume is related to the extent of uncertainty (or error) in the state estimate, a diverging covariance could therefore be linked to a diverging estimation error, and furthermore to the divergence of two points in the system (the true state and the estimated state). Thus, estimating the Lyapunov exponents of the estimation error may help give an indication as to which objects will have the greatest tendency towards these diverging estimation errors.

An illustration of how these estimation errors may diverge over time in the absence of observations (in this case, reflected by the relative distance between two objects with very close initial conditions) for a low Earth orbit (LEO), medium Earth orbit (MEO) and geosynchronous Earth orbit (GEO) orbit is seen in Figure 4.1. In this figure, it is observed that the two initially close LEO orbits have the greatest final separation while the GEO orbits have the least, indicating that when calculating Lyapunov exponents for each of these orbits, the size and shape of the orbit should in part dictate the divergence of the estimation error. The larger

displacements for the LEO orbit also indicates that a small estimation error for a LEO object will turn into a large error at a quicker rate than a GEO object, and therefore a LEO object should theoretically be observed more frequently in a sensor schedule than a GEO object.

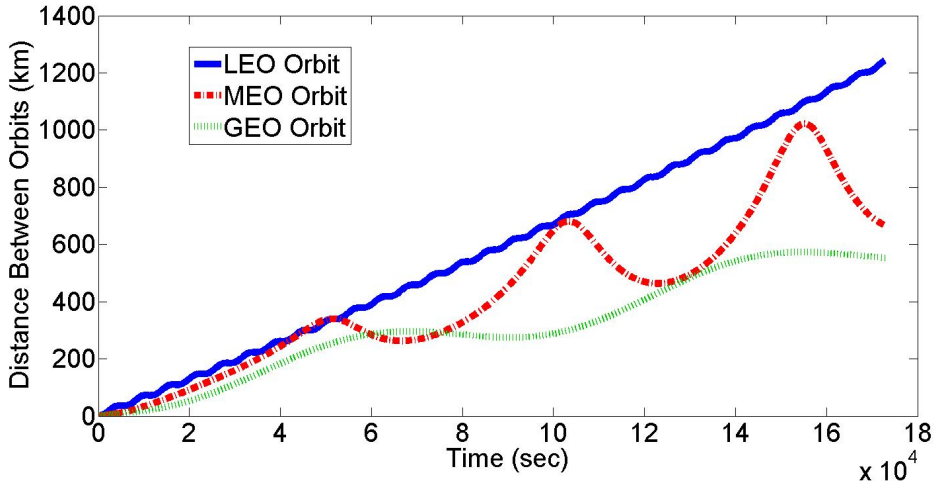

Figure 4.1. Plots of the propagation of displacement between two initially close LEO, MEO and GEO orbits.

While no deterministic system exists for the calculation of the Lyapunov spectrum, the work of Wolf et. al. [37] provided a convenient framework for calculating this largest Lyapunov exponent from experimental data, through a fixed evolution time program for λ^1

$$\lambda^1_{i,k+1} = \frac{1}{t_{k+1} - t_0} \sum_{k'=0}^{t_k/\Delta t} log_2 \frac{D'_{i,k'+1}}{D_{i,k'}} \tag{4.87}$$

where the term $D_{i,k'}$ refers to the distance (in a Euclidean sense) between a point on a reference trajectory (a trajectory in state space to measure divergence with respect to) and its closest data point at time $t_{k'} = k'\Delta t$, while $D'_{i,k'+1}$ is the propagation of that distance to the time $t_{k'+1} = (k'+1)\Delta t$. In this formulation, the closest data point must be determined at each time step. Equation 4.87 is reformulated for these studies to account for the constant time increments of Δt when measurements are possible (but not necessarily taken), and objects are indexed by the subscript i. Additionally, for these studies, t_0 reflects the initial time

of the simulation, which will always be defined by $t_0 \equiv 0$.

Using this as a benchmark, Rauf et. al.[38] determined how an *effective Lyapunov exponent* λ_{eff} could be determined from a nonlinear adaptive filter utilizing the root mean square (RMS) error determined from the filters estimates. This process essentially modifies Eq. 4.84 such that the distance measurements $\delta_{t+\Delta t}$ and δ_t are given from the RMS error E_T^{RMS} and E_0^{RMS} between some initial time t_0 and final time t_f, for the purpose of calculating an estimate to $\lambda^1 \approx \lambda_{eff}$.

For these studies, this approach of using the RMS error as the distance measurements to estimate the largest Lyapunov exponent is applied to Eq. 4.87 such that $D_{i,k'} = E_{i,k'}^{RMS}$ and $D'_{i,k'+1} = E_{i,k'+1}^{RMS}$. This suggested process could be viewed as applying Eq. 4.87 to a single-point data set to yield a very basic estimate for the largest Lyapunov exponent

$$\hat{\lambda}_{i,k+1}^1 = \frac{1}{t_{k+1}} \sum_{k'=0}^{t_k/\Delta t} log_2 \frac{E_{i,k'+1}^{RMS}}{E_{i,k'}^{RMS}} = \frac{1}{t_{k+1}} log_2 \frac{E_{i,k+1}^{RMS}}{E_{i,0}^{RMS}} \tag{4.88}$$

Furthermore, since the filters used in these problems provide covariance estimates at the time when tasking occurs (the prediction step) $\hat{P}_{i,k+1}^f$, an estimate for the RMS error of a particular objects state estimate at a particular time step $k+1$ can be calculated from $E_{i,k+1}^{RMS} = \sqrt{\text{trace}\left(\hat{P}_{i,k+1}^f\right)}$, and from an initial time step $k = 0$ by $E_{i,0}^{RSME} = \sqrt{\text{trace}\left(\hat{P}_{i,0}^*\right)}$. Through application of these calculations for the estimated RMS error, Eq. 4.88 becomes

$$\hat{\lambda}_{i,k+1}^1 = \frac{1}{t_{k+1}} log_2 \sqrt{\frac{\text{trace}\left(\hat{P}_{i,k+1}^f\right)}{\text{trace}\left(\hat{P}_{i,0}^*\right)}} \tag{4.89}$$

Of particular note in regards to Eq. 4.89 is that it represents a way to determine the estimation error stability of a particular object in the proposed tracking problem, which was motivated by concepts of Lyapunov exponent estimation. However, due to the fact that it is merely an approximation, and that Lyapunov exponents are typically calculated using very large data sets (and this method is only using

the time history of one data point), these methods should be viewed as providing a useful utility metric for tasking rather than a competitive method for accurately determining a largest Lyapunov exponent.

An additional convenience in using Eq. 4.89 is that is can be positive or negative, depending on how the uncertainty has increased or decreased over the entire simulation. This allows for objects with consistent covariance estimates to have near zero-mean values for $\hat{\lambda}^1_{i,k+1}$, and objects whose covariances are decreasing to have negative values of $\hat{\lambda}^1_{i,k+1}$. However, due to the diverging nature of the propagation of the covariance estimate in the prediction step for both an EKF and UKF, it follows that unobserved objects will have increasing values of $\hat{\lambda}^1_{i,k+1}$. Also, if the value of $\hat{\lambda}^1_{i,k+1}$ remains high for a particular object which is sparsely available for observations, it simply makes that object a priority, in that its value for $\hat{\lambda}^1_{i,k+1}$ is so high with respect to others that it may be viewed at every opportunity to take a measurement, with those opportunities occurring rarely. This concept will be highlighted later in Chapter 5.

Thus, given a motivation for each sensor to observe the object whose history trends strongest towards divergence, it follows that when using an LLE-based method of tasking, the elements of the visibility matrix can be evaluated as

$$\mu_{(i,j),k+1} = \hat{\lambda}^1_{i,k+1} \forall \, [i,j] \in \left[\mathcal{O}^a_{k+1}, \mathcal{S}^a_{i,k+1}\right] \tag{4.90}$$

Additionally, the decision process can be very simply represented as each sensor observes the object with the highest largest Lyapunov exponent estimate, given all the available objects to that sensor.

$$\xi_{(i,j),k+1} = \begin{cases} 1 & i = \max\limits_{i} \mu_{(i,j),k+1}, \;\; \mu_{(i,j),k+1} \neq 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.91}$$

4.4.1 Calculating the LLE for an EKF

Evaluating the LLE for an EKF consists of merely using the initial covariance estimate for object i $\hat{P}^*_{i,0}$ (which is equal for all objects in all simulations, as described

in Chapter 6), and the forecast covariance estimate evaluated using Eq. 3.22 as inputs in Eq. 4.89. It should be noted that the LLE is based only on the object, and has no dependence on the sensor(s) which are available to that object.

4.4.2 Calculating the LLE for a UKF

Evaluating the LLE for an EKF consists of merely using the initial covariance estimate for object i $\hat{P}^{*}_{i,0}$ (which is equal for all objects in all simulations, as described in Chapter 6), and the forecast covariance estimate evaluated using Eq. 3.35 as inputs in Eq. 4.89. It should be noted that the LLE is based only on the object, and has no dependence on the sensor(s) which are available to that object.

4.4.3 Calculating the LLE for an AEGIS Filter

Evaluating the LLE for an EKF consists of merely using the initial covariance estimate for object i $\hat{P}^{*}_{i,0}$ (which is equal for all objects in all simulations, as described in Chapter 6), and the forecast covariance estimate evaluated using Eq. 3.60 (replacing the superscript $*$ with f, serving as an approximation $\hat{P}^{f,gmm}_{i,k+1} \approx \hat{P}^{f}_{i,k+1}$) as inputs in Eq. 4.89. It should be noted that the LLE is based only on the object, and has no dependence on the sensor(s) which are available to that object.

Chapter 5

Simulation Results

5.1 Initialization

Using the nonlinear dynamics and measurement functions outlined in Eqs. 2.2 and 2.15, the necessary inputs to execute the estimation and tasking outlined in Chapters 2-4 are presented. Two types of simulations will be executed. The first will be a simulation in which initial state estimate errors, covariance estimates, and sensor noise reflect values similar to what can be expected for the tracking of space objects, and is referred to as the *low-error* test case [29]. The second simulation, referred to as the *high-error* test case, will have slightly higher initial position errors and sensor noise, to determine if coupling effects are dramaticized as filters are stressed.

The initial estimates for each objects state and covariance are provided, given by

$$\hat{X}^*_{i,0} = \vec{x}_{i,0} + \Delta\hat{X}_i \tag{5.1}$$

where $\Delta\hat{X}_i$ represents a value generated from a zero-mean Gaussian distribution of covariance $\hat{P}^*_{i,0}$ and PDF defined as $p^g\left(\vec{x}_{i,0}; 0, \hat{P}^*_{i,0}\right)$. Additionally, the initial covariance is defined by

$$\hat{P}^{*}_{i,0} = \begin{bmatrix} \left(\hat{\sigma}^{x}_{i,0}\right)^2 & 0 & 0 & 0 \\ 0 & \left(\hat{\sigma}^{y}_{i,0}\right)^2 & 0 & 0 \\ 0 & 0 & \left(\hat{\sigma}^{\dot{x}}_{i,0}\right)^2 & 0 \\ 0 & 0 & 0 & \left(\hat{\sigma}^{\dot{y}}_{i,0}\right)^2 \end{bmatrix} \tag{5.2}$$

Initial conditions for the states of the sensors as well as their field of regard constraints are found in Table 5.1. The state trajectories resulting from these initial conditions are generated using a $7^{th} - 8^{th}$ order Runga-Kutta numerical integration algorithm. Possible sensor measurements are then generated when objects are in the field of regard of a sensor via the measurement Eq. 2.15 applied to these trajectories.

Four ground-based sensors are chosen so that their respective field of regard covers the majority of space around low Earth orbit (LEO), with that area dissipating as objects get closer to geosynchronous Earth orbit (GEO). To simulate this, each of these four sensors can view objects in LEO, two can view objects in medium Earth orbit (MEO) and one can view objects in GEO. This was done so that the sensor-object dynamics would have some diversity, encompassing many orbits, and varying opportunities for observation, as well as periods where no observations were possible. Additionally, a single orbiting sensor is introduced to allow for multiple sensors to view a single object, as well as cover a large orbit regime (i.e. the orbiting sensor has the possibility to make observations in LEO, MEO, or GEO, but not all three at one time). A table containing sensor range and half-angle constraints are presented in Table 5.2.

Table 5.1. SSA Sensor Initial States (note $j = 1$ represents orbiting sensor)

Sensor (j)	$x^{s}_{j,0}$ (km)	$y^{s}_{j,0}$ (km)	$\dot{x}^{s}_{j,0}$ (km/sec)	$\dot{y}^{s}_{j,0}$ (km/sec)
1	3.9138×10^3	6.8712×10^3	-8.2702	4.0600
2	4.1925×10^3	4.7972×10^3	-0.3498	0.3057
3	-3.5957×10^3	5.2594×10^3	-0.3835	-0.2622
4	5.7576×10^3	-2.7277×10^3	0.1989	0.4198
5	-4.9047×10^3	-4.0662×10^3	0.2965	-0.3577

Furthermore, the constant additive zero-mean Gaussian process noise and its

Table 5.2. SSA Sensor Constraints (note $j = 1$ represents orbiting sensor)

Sensor (j)	Δ_j (km)	Ψ_j (deg)
1	10^4	180.0
2	4.4157×10^4	10.0
3	2.5371×10^4	20.0
4	8.871×10^3	50.0
5	3.0×10^4	15.0

associated covariance are given by

$$\vec{w}_i = \begin{bmatrix} 0 \\ 0 \\ w_{\ddot{x}} \\ w_{\ddot{y}} \end{bmatrix} \tag{5.3}$$

$$Q_i = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & w_{\ddot{x}}^2 & 0 \\ 0 & 0 & 0 & w_{\ddot{y}}^2 \end{bmatrix} \tag{5.4}$$

Table 5.3. Constants in SSA Simulation

Constant	Value
μ_E	3.986×10^5 (km^2/sec^3)
ω_E	7.2921×10^{-5} (rad/sec)
$w_{\ddot{x}}$	10^{-6} (km/sec^2)
$w_{\ddot{y}}$	10^{-6} (km/sec^2)
t_f	$172,328$ (sec)
$t_f/\Delta t$	500

All other constants not specific to either the low-error or high-error simulations are given in Table 5.3. Constants for the simulation time and number of time steps are chosen so that objects can experience a range of estimate qualities based on an object's number of observations, and time gaps between possible observations. The purpose of this is to highlight penalties/benefits of tasking decisions. As stated previously, the primary objective of these studies is to investigate the coupling between tasking and estimation using satellite tracking as an example, and not to exactly model the real-world process of monitoring space objects (such as in space situational awareness, SSA). Therefore, these constants were chosen so they could

best aid in this investigation.

5.2 Post-Simulation Performance Metrics

Once a simulation is completed, various performance metrics are calculated to give a quantitative measure of how well each estimator-tasking combination tracked all N_s objects throughout the duration of the simulation. These performance metrics should be chosen so that insights can be gained not only on how well the state estimates performed with respect to the true state $\vec{x}$, but also how well the covariance estimates reflected the error between the estimated and true state. Performance metrics should also highlight how estimation errors were distributed (for example, were there many objects with low errors and only a few with high, or were the errors evenly distributed) allowing for a more in-depth assessment of performance.

Candidates for scalar performance metrics (to give a top-level snapshot of performance) include one to measure the overall estimation accuracy of the simulation, which is the average position error over all objects for the simulation time span

$$\langle E^r \rangle = \frac{1}{N_s \left(t_f/\Delta t\right)} \sum_{i=1}^{N_s} \sum_{k=0}^{t_f/\Delta t} \Delta r_{i,k} \tag{5.5}$$

where

$$\Delta r_{i,k} = \sqrt{\left(x_{i,k} - \hat{x}^*_{i,k}\right)^2 + \left(y_{i,k} - \hat{y}^*_{i,k}\right)^2} \tag{5.6}$$

Additionally, the average position error for each object can be calculated by

$$\langle E^r_i \rangle = \sum_{k=0}^{t_f/\Delta t} \frac{\Delta r_{i,k}}{\left(t_f/\Delta t\right)} \tag{5.7}$$

Another scalar performance metric is also calculated in order to estimate difficulty with data association, which is average estimated error ellipse area

$$\left\langle \hat{A}_{err} \right\rangle = \frac{1}{N_s \left(t_f/\Delta t\right)} \sum_{i=1}^{N_s} \sum_{k=0}^{t_f/\Delta t} \pi \sqrt{\hat{\varsigma}_{i,k}^1 \hat{\varsigma}_{i,k}^2} \tag{5.8}$$

where $\hat{\varsigma}_{i,k}^1$ and $\hat{\varsigma}_{i,k}^2$ represent the eigenvalues of the upper left 2×2 matrix of the covariance estimate (or the covariance of the position estimates). Assuming that these eigenvalues represent an estimated semi major and semi minor axis of a Gaussian error ellipse, the metric given in Eq. 5.8 will give an indication of difficulty in data association due to the fact that as estimated errors increase, the area of uncertainty surrounding a particular state estimate will grow, making pinpointing an object's exact location, as well as possibly differentiating it amongst other neighboring satellites more difficult.

Graphical performance metrics were also calculated to give a visual representation of how each estimator/tasking combination performed. The first of which is simply a histogram showing the percentage of position errors falling within error bounds of $\Delta r_{i,k} < 1$, $1 < \Delta r_{i,k} < 5$, $5 < \Delta r_{i,k} < 10$, up to $\Delta r_{i,k} > 1000$ where all units are in kilometers.

Also, histograms are created which give an indication on how well the estimated position errors, provided by the filters were performing with respect to the true position errors,

$$J_{i,k}^{\hat{P}} = \frac{\Delta r_{i,k}}{\sqrt{\left(\hat{\sigma}_{i,k}^x\right)^2 + \left(\hat{\sigma}_{i,k}^y\right)^2}} \tag{5.9}$$

This metric is calculated for each object at each simulation time step, and plotted as a histogram binning them in 0.2 increments from 0 to 3, where each incriment represents a multiplicative factor of the estimated position standard deviation $\hat{\sigma}_{i,k}^r = \sqrt{\left(\hat{\sigma}_{i,k}^x\right)^2 + \left(\hat{\sigma}_{i,k}^y\right)^2}$. This is done to show how close the actual position error is to the estimated position standard deviation $\left(\hat{\sigma}_{i,k}^r = \sqrt{\left(\hat{\sigma}_{i,k}^x\right)^2 + \left(\hat{\sigma}_{i,k}^y\right)^2}\right)$ derived from the covariance estimate[1].

[1] A value of $J_{i,k}^{\hat{P}} = 1$ would represent a position error reflecting exactly one estimated position standard deviation

The true state at each time step, $\vec{x}_{i,k}$ is calculated before running the simulation by numerically integrating Eq. 2.2 over the time span $t \in [0, t_f]$ using a variable step 7^{th} - 8^{th} order Runga-Kutta algorithm. These metrics were not used in the simulation (except for data association, which was removed as a variable in these studies as described in Chapter 2), but were simply used to quantify the performance of the respective filtering algorithms.

5.3 Low-Error Simulation

As described in Chapter 2, 100 objects are distributed with semi major axes and eccentricities shown in Figure 5.1. Even though 100 objects were originally created, only 94 of these objects made passes within at least one sensor's field of regard during the simulation time span. Since there is no purpose in evaluating performance of objects which could never have been physically observed by any sensors, all simulation performance is calculated with respect to the $N_s = 94$ objects which had the possibility for observation.

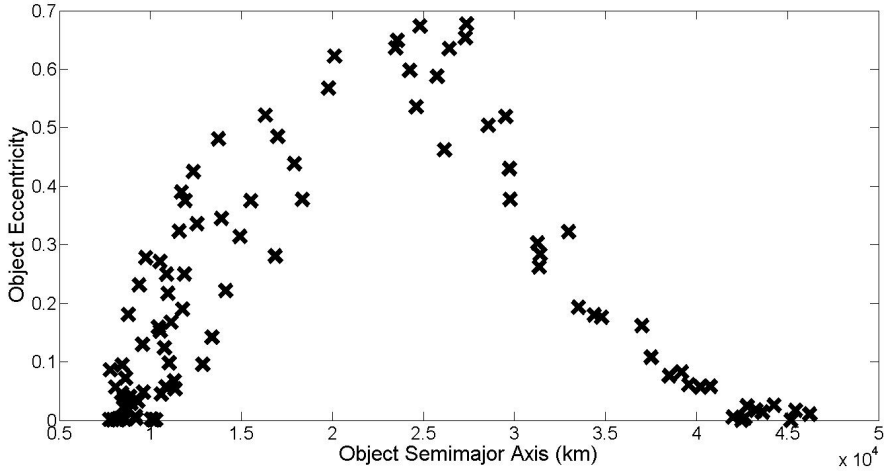

Figure 5.1. Distribution of semimajor axes and eccentricities of 100 objects in low-error simulation

Initial object state standard deviations (used in both selecting the initial state

estimates, and defining the initial covariance estimate) for the low-error test case are given in Table 5.4,

Table 5.4. Initial Standard Deviations: Low-Error Simulation

Standard Deviation	Value
$\hat{\sigma}^x_{i,0}$	1 (km) $\forall i$
$\hat{\sigma}^y_{i,0}$	1 (km) $\forall i$
$\hat{\sigma}^{\dot{x}}_{i,0}$	10^{-3} (km/sec) $\forall i$
$\hat{\sigma}^{\dot{y}}_{i,0}$	10^{-3} (km/sec) $\forall i$

Sensor noise for these low-error simulations are chosen to reflect values close to the ideal range and angle variances used in satellite tracking applications[29] with $\left(v^{\rho}_j\right)^2 = 0.5$ km^2 and $\left(v^{\psi}_j\right)^2 = 0.1$ deg^2 $\forall j$.

Using these inputs and the performance metrics outlined in Section 5.2, simulations for 94 satellites, four ground based sensors and one space based sensor are conducted by implementing FIG, SIG, and LLE-based tasking utilizing EKF, UKF, and AEGIS estimators. However, as an initial comparison, performance metrics outlined in Section 5.2 are presented for an ideal tracking scenario in which sensors can measure all objects in the field of regard at each time step. This ideal scenario, referred to henceforth as the *all data* case, represents an estimate for the lower bound on the performance of any case where the number of simultaneous measurements is limited. In the following sections, performance is evaluated for the ideal all data case, followed by a further investigation into the FIG, SIG, and LLE-based tasking strategies.

5.3.1 All Data Tasking

Histograms showing percentage of position errors falling within the bins described in chapter 5 for the all data simulation is presented in Figure 5.2. Another histogram showing the percentage of position errors falling within specified multiplicative factors of the estimated position standard deviation are found in Figure 5.3. Additionally, tabulated results showing the scalar performance metrics outlined in Eqs. 5.5 and 5.8 are presented in Table 5.5.

Table 5.5. Simulation Performance Metrics for All Data Low-Error Simulation

Filter	$\langle E_r^* \rangle$ (km)	$\left\langle \hat{A}_{err} \right\rangle$ (km^2)
EKF	5.512	3.630×10^2
UKF	5.515	2.773×10^2
AEGIS	4.632	2.570×10^2

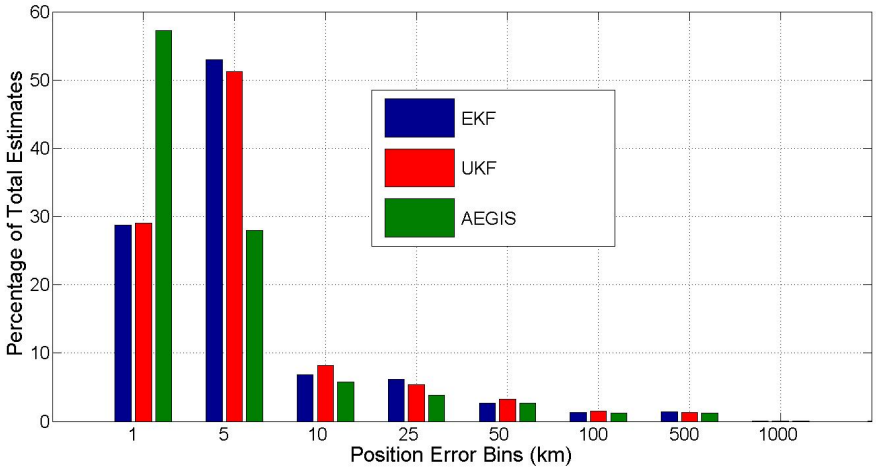

Figure 5.2. EKF, UKF, and AEGIS histograms of position estimate error distributions for all data simulation. Plots include data from all objects at all simulation time steps.

From observing Figures 5.2 - 5.3 as well as Table 5.5, the AEGIS filter has the best performance in the all data simulation, followed by the UKF and EKF. The AEGIS performs the best due to the fact that it had the lowest average position error (resulting from the greatest percentage of position estimates within 1 km), as well as the lowest average estimated error ellipse area. Figure 5.3 also shows that while the AEGIS filter provides the lowest average estimated error ellipse area, it does so at the cost of occasionally having overconfident uncertainty estimates (as shown by having the greatest percentage of position errors $\Delta r_{i,k} \geq \hat{\sigma}_{i,k}^r$). An overconfident uncertainty estimate implies that the estimated uncertainty (or standard deviation) derived from the covariance estimate may be too low with respect to the actual error of the state estimate. However, since the AEGIS filter produced more position errors are within $0.8\hat{\sigma}_{i,k}^r \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}_{i,k}^r$ than the EKF or UKF, it also produced the most accurate estimates. Using similar logic, it is easy to see that the EKF and UKF filters provide a large amount of position errors $\Delta r_{i,k} \leq 0.8\hat{\sigma}_{i,k}^r$, implying that just as the AEGIS filter may have the tendency to be overconfident,

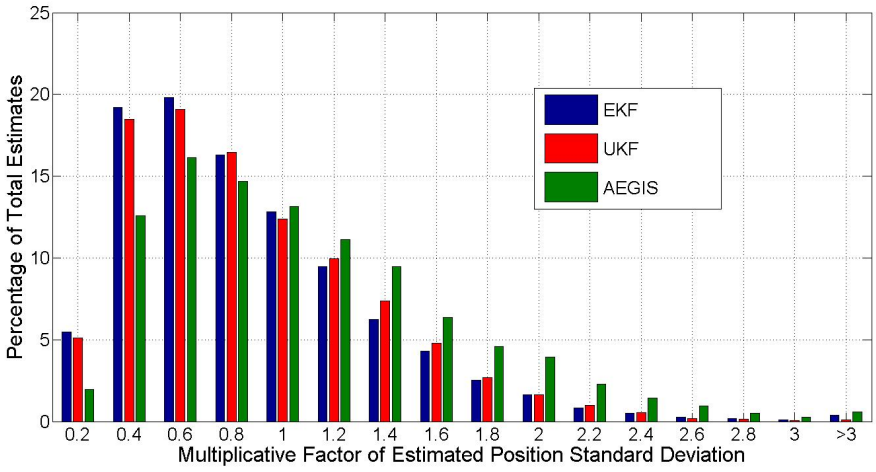

Figure 5.3. EKF, UKF, and AEGIS histograms of $J_{i,k}^{\hat{P}}$ metric in Eq. 5.9 for all data simulation. Plots include data from all objects at all simulation time steps

the EKF and UKF may be under confident (i.e. producing uncertainty estimates that are too high compared to the actual estimate error).

While Table 5.5 shows the UKF and EKF have similar performance with respect to their average position errors, the UKF provides these estimates with a lower average position error ellipse than the EKF. The significance is in the propagation and updating of the covariance estimates, where it is shown here that the UKF can provide similar state estimate quality to the EKF but to less of a degree of uncertainty. Furthermore, the AEGIS filter provides an even lower degree of uncertainty in its estimates. To investigate why the UKF and AEGIS filter provide better uncertainty estimates than the EKF, Figure 5.4 shows the propagation of the average estimated error ellipse area in Eq. 5.8 for the object with the worst average position error. This figure illustrates an important fact that despite having identical measurement data at identical times, the UKF, and furthermore the AEGIS filter provide not only better propagation of uncertainty (illustrated from the timespan 0 - 110,000 sec), but also updates in uncertainty (highlighted from the timespan 110,000 - 130,000 sec). These reflect advantages the UKF filter has in the nonlinear estimation of uncertainty over the EKF, but also in the AEGIS filter's advantage in updating uncertainty estimates using a GMM over the Gaussian uncertainty of the UKF. Advantages such as these in the propagating

and updating of uncertainty and/or state estimates (though uncertainty will have more sway over the performance when coupled with tasking, since all tasking metrics are covariance-based) will play a critical role in the performance advantages these filters will gain when coupled with a covariance-based tasking strategy.

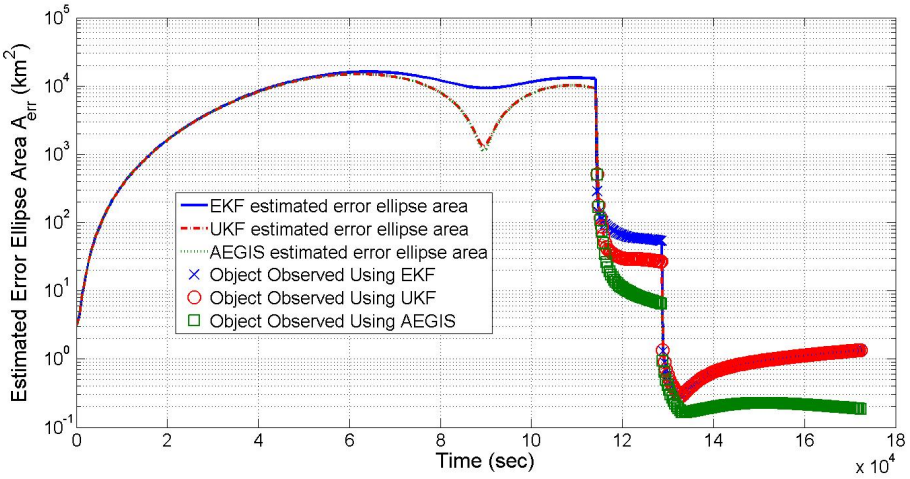

Figure 5.4. EKF, UKF, and AEGIS plots of $\hat{A}_{err} = \pi\sqrt{\hat{\varsigma}^1_{i,k}\hat{\varsigma}^2_{i,k}}$ metric for the object with the highest average position error in the all data simulation.

To investigate how each individual object performed in the all data simulation, Figure 5.5 shows the average position error for each simulated object, while Figure 5.6 shows how many updates each of these objects received. Figure 5.6 illustrates the diversity in the number of updates each object received, which Figure 5.5 shows is strongly correlated to the average position error an object would obtain throughout the simulation. Even in this ideal case where sensors can observe more than one object at a time, some objects are sparsely available for observation (e.g. objects 18, 38, 58, and 75), while other objects have several (most notably object 57, which is observable by at least one sensor at every simulation time step). For all objects, the AEGIS filter had the greatest tendency to provide the best average position errors as opposed to the EKF or UKF, which both had a series of objects with the worst errors.

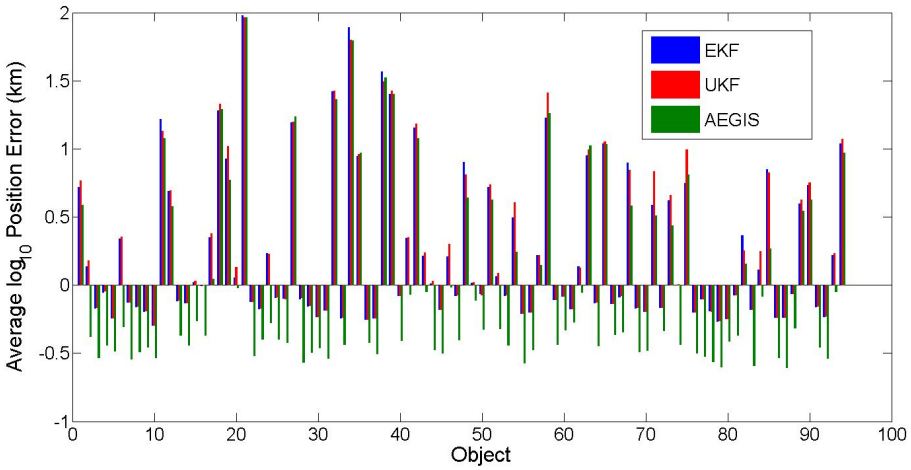

Figure 5.5. Bar graphs showing the $\log_{10}$ average position error for each object given in Eq. 5.7, through implementation of an EKF, UKF, and AEGIS filter in conjunction with all data tasking strategy. In this case, all filters have equal amounts of updates for each object, reflecting the maximum updates possible for all objects.

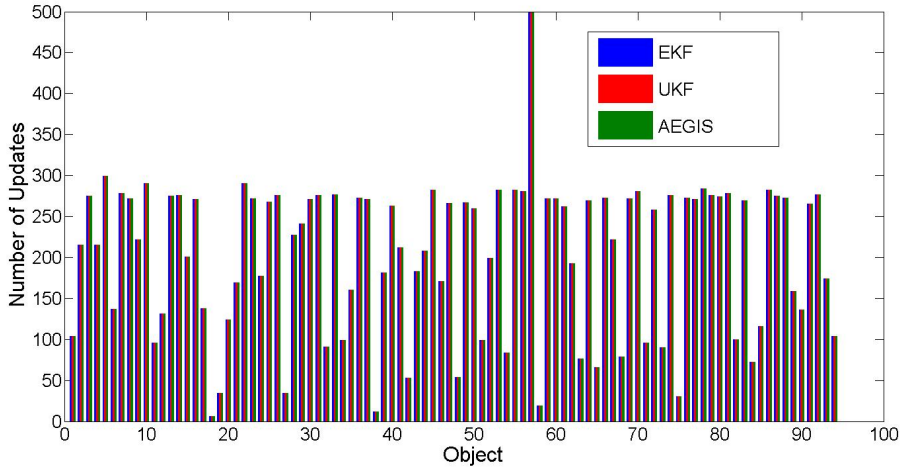

Figure 5.6. Bar graphs showing the updates each object received through implementation of an EKF, UKF, and AEGIS filter in conjunction with all data tasking strategy. In this case, all filters have equal amounts of updates for each object, reflecting the maximum updates possible for all objects.

5.3.2 FIG Tasking

Table 5.6 shows the scalar performance metrics outlined in Eqs. 5.5 and 5.8 for the low-error simulation using FIG tasking. In addition, figures showing the distribution of position errors and how these errors compared to the estimated $3\hat{\sigma}^r_{i,k}$ provided by the filter covariance estimates are presented in Figures 5.7 - 5.8. Results show that the EKF is especially affected by this selection of tasking strategy, as opposed to the UKF or AEGIS filter. This is shown in the very large average error metric in Table 5.6, along with a drastic increase of position estimate errors ($\Delta r_{i,k}$) greater than 25 km in Figure 5.7, as well as percentage of position estimate errors greater than $3\hat{\sigma}^r_{i,k}$ in Figure 5.8. In the latter case, a position estimate error greater than $3\hat{\sigma}^r_{i,k}$ implies that the actual position error is much greater than what the filter's estimates are indicating they should be. This is also reflected in the discrepancy of the values for $\left\langle \hat{A}_{err} \right\rangle$ between the EKF and the other filters in comparison to its much larger deviations in $\langle E^*_r \rangle$. This is most evident when observing the 3^{rd} and 4^{th} columns in Table 5.6, where the errors accumulated in the FIG simulation are normalized with respect to the errors those filters achieved in the all data simulation. These results show that the EKF produced a position error over 100 times larger using FIG tasking as opposed to the UKF and AEGIS filters which were marginally worse by comparison. Also, the EKF produced an estimated error ellipse area only 5 times greater than in the all data simulation, implying that the EKF has a greater tendency to underestimate the possible error in its estimations than the UKF or AEGIS filter. The poor performance shown in the EKF-FIG combination would lead to either severely inaccurate object uncertainty and/or location estimates, or an inability to continue to monitor those objects should these methods be used in a real satellite tracking application.

As for the UKF-FIG combination, the performance was much better, and did not experience the issues that were present in the implementation of the EKF. In particular, the UKF produced very little position errors outside of the $3\hat{\sigma}^r_{i,k}$ bounds as shown in Figure 5.8. This is the result of the UKF being able to produce a Gaussian covariance which more accurately models the actual uncertainty in an object's estimates than the EKF, an effect which was highlighted in Figures 3.7 - 3.8.

The AEGIS filter produced the best performance using the FIG tasking, as reflected by having the lowest scalar performance metrics in Table 5.6. Furthermore, Figure 5.7 shows that the AEGIS filter provided a disproportionate amount of estimates within the tightest position error bounds of $\Delta r_{i,k} \leq 1$ km, as it did in the all data simulation. When observing Figure 5.8, while the AEGIS filter produced more position errors outside of the $3\hat{\sigma}^r_{i,k}$ bounds, these estimates were few (roughly 1%). What is more important to realize in these results is that the AEGIS filter produced the most position errors within $0.8\hat{\sigma}^r_{i,k} \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}^r_{i,k}$, meaning that while it produced slightly more position errors outside the $3\hat{\sigma}^r_{i,k}$ bounds than the UKF, it also generally produced more accurate uncertainty estimates than the UKF or EKF.

Table 5.6. Simulation Performance Metrics for FIG Low-Error Simulation

Filter	$\langle E^*_r \rangle$ (km)	$\left\langle \hat{A}_{err} \right\rangle$ (km^2)	$\frac{\langle E^*_r \rangle_{Tasking}}{\langle E^*_r \rangle_{All\ Data}}$	$\frac{\left\langle \hat{A}_{err} \right\rangle_{Tasking}}{\left\langle \hat{A}_{err} \right\rangle_{All\ Data}}$
EKF	5.627×10^2	1.806×10^3	102.086	4.975
UKF	7.950	3.202×10^2	1.442	1.155
AEGIS	7.290	2.809×10^2	1.574	1.093

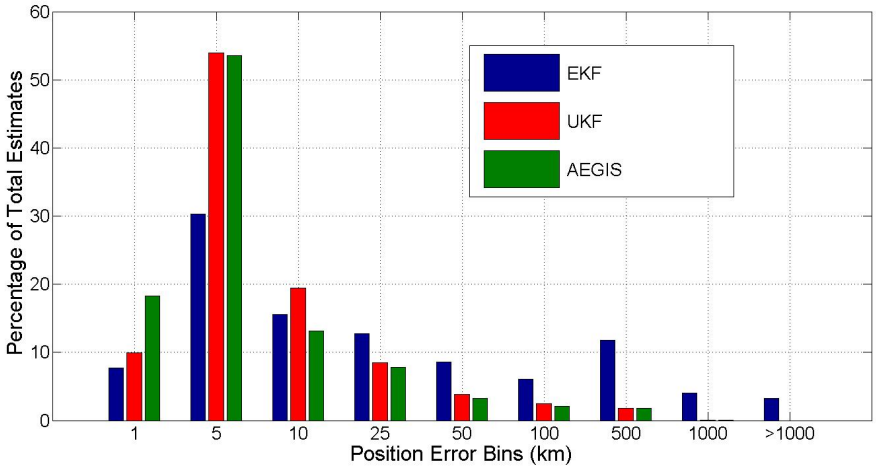

Figure 5.7. EKF, UKF, and AEGIS histograms of position estimate error distributions for FIG simulation. Plots include data from all objects at all simulation time steps.

To investigate why performance discrepancies existed in the implementation of

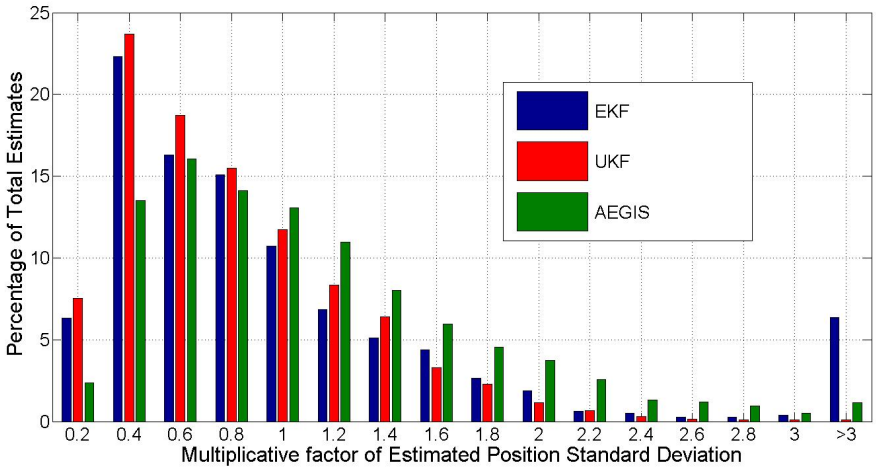

Figure 5.8. EKF, UKF, and AEGIS histograms of $J^{\hat{P}}_{i,k}$ metric in Eq. 5.9 for FIG simulation. Plots include data from all objects at all simulation time steps

EKF, UKF, and AEGIS filters in conjunction with an FIG tasking strategy, Figures 5.9 - 5.10 show the amount of updates each simulated object received along with their corresponding average position errors in the FIG simulation.

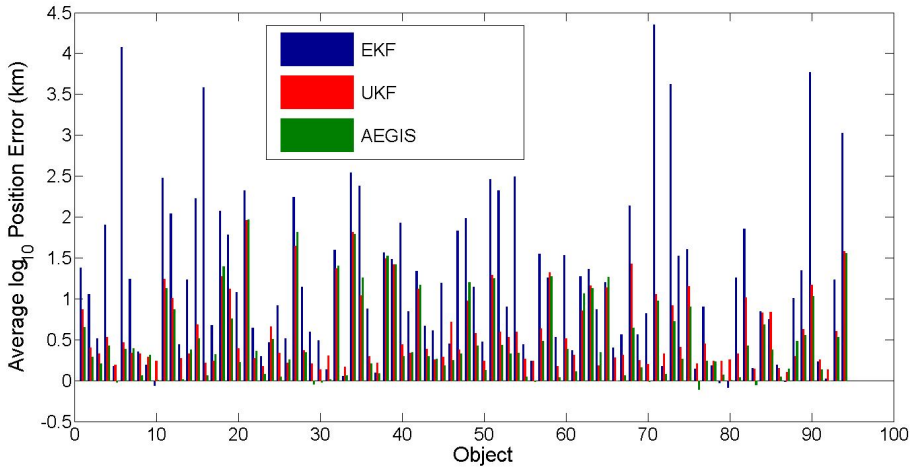

Figure 5.9. Bar graphs showing the $\log_{10}$ average position error for each object given in Eq. 5.7, through implementation of an EKF, UKF and AEGIS filter in conjunction with an FIG tasking strategy.

As shown in Figure 5.10, when using FIG tasking, several objects received a disproportionately large amount of updates (e.g. objects 10, 37, 92, etc.) with

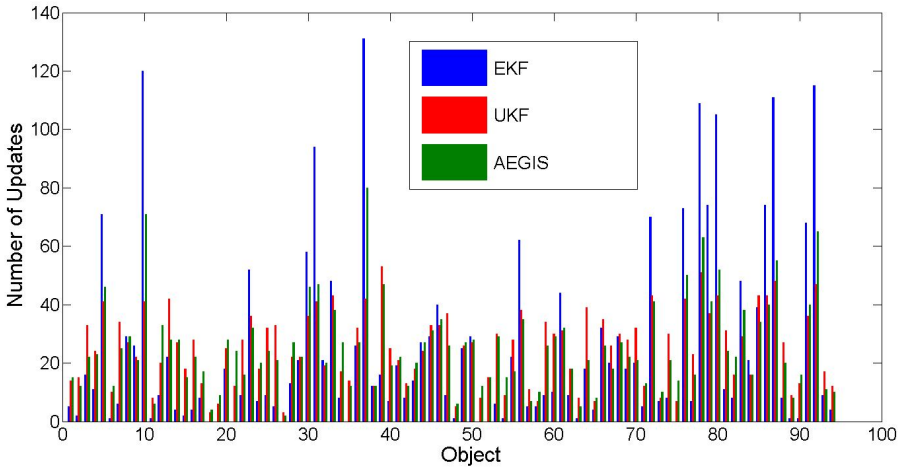

Figure 5.10. Bar graphs showing the updates each object received through implementation of an EKF, UKF, and AEGIS filter in conjunction with an FIG tasking strategy. The FIG tasking results in many updates to a select few objects, while many other objects receive little to none.

respect to many others which received little to none (e.g. objects 11, 18, 27, etc.). The result is that many objects accumulated large estimate errors due to these few updates, while those that had the most observations resulted in much better average position estimate errors. In particular, the objects which received little to no observations had errors which were much greater when using an EKF as opposed to a UKF or AEGIS filter, which resulted in the vast performance discrepancies as seen in Table 5.6, as well as Figures 5.7 - 5.8. This was in part due to the fact that, as illustrated in the work of Teixeira et. al.[28], less updates will result in longer propagation times for the estimates which will negatively affect the EKF covariance estimates more than the UKF or AEGIS filter due to its linear covariance propagation. Furthermore, Figure 5.10 shows that for the particular objects that received little updates, the UKF and AEGIS filters consistently had more updates than the EKF (which sometimes had zero). These two factors contributed to several objects having lower average position estimate errors when using a UKF or AEGIS filter as opposed to an EKF. Similarly, the EKF always produced more observations for many objects which had the lowest average position estimate errors (e.g. objects 10, 31, 37, etc.), which generally resulted in those objects having comparable average position estimate errors using an EKF with respect to a UKF

or AEGIS filter (commonly, the EKF would produce lower errors than the UKF in many of these cases). However, these few good objects for the EKF could not compare to the numerous ones which performed poorly, resulting in the UKF and AEGIS filters severely outperforming the EKF for the application of FIG tasking.

While the UKF and AEGIS filters provided better performance for many objects as opposed to the EKF, discrepancies were still present between the two. Specifically, for a majority of objects the AEGIS filter would produce average errors lower than the UKF, while also dispensing observations differently than the UKF. For example, the AEGIS filter would generally provide slightly more updates to objects which were either available for few observations (e.g. objects 18, 58, 75), or objects which had several observations and generally low errors (e.g. objects 5, 10, 37, etc.). In each of these cases the AEGIS filter produced equal or better average position errors than the UKF. When coupled with the fact that the AEGIS filter generally produced comparable (often better) average position errors for objects for which it distributed less updates than the UKF, the result is a more efficient use of tasking decisions.

To investigate why there was such a performance discrepancy between the EKF and the UKF/AEGIS filters when implementing the FIG metric, Figures 5.11 - 5.16 show when object 71 (the worst performing FIG object using either filter) was available for observation, tasked for observation, as well as the time histories of its position estimate error using FIG, the position error in the all data case, its maximum FIG utility metric (at times it was available for observation), and the average FIG metric for objects tasked for observation. Figure 5.14 shows that despite being available for observation several times during the simulation, using the EKF only resulted in 5 observations while the UKF and AEGIS filter resulted in 12 and 13 observations respectively. This increase in observations results in the position error for this object being much better, and more stable for the UKF and AEGIS filters than the EKF, where it diverged to the point where the filter failed to provide realistic estimates. If the EKF-FIG combination instead resulted in the object being tasked for observation early in the simulation, as was the case with the other filters, this divergence may have been avoided.

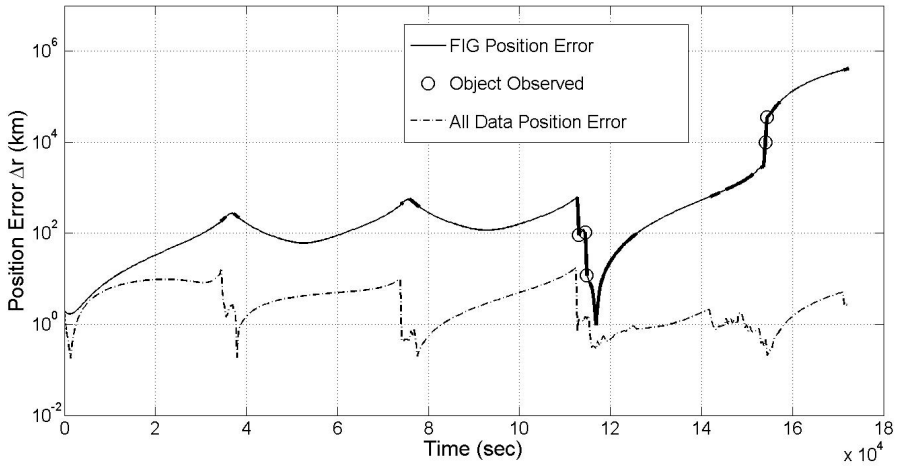

Figure 5.11. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 71 using FIG tasking strategy in conjunction with an EKF. Thick sections of line represent times when observations were available between object 71 and one or more sensors.

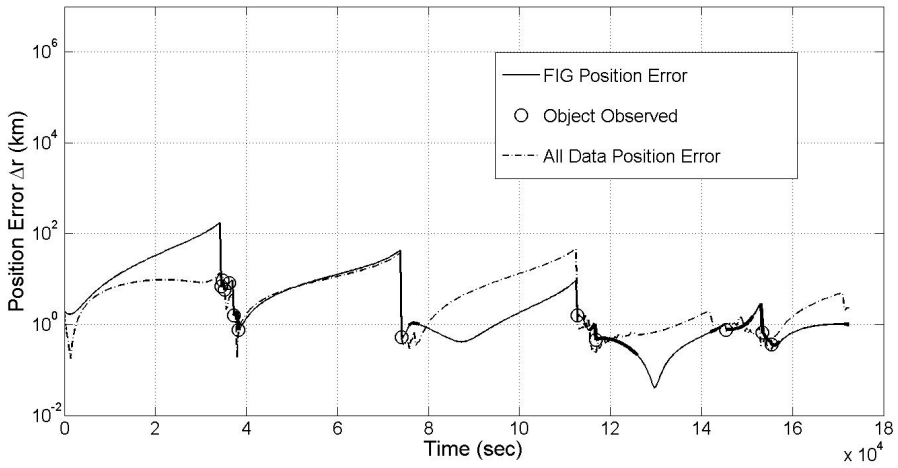

Figure 5.12. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 71 using FIG tasking strategy in conjunction with a UKF. Thick sections of line represent times when observations were available between object 71 and one or more sensors.

When observing Figure 5.14, it is easy to see why this occurred. The EKF-based FIG metric for object 71 consistently stayed below the average FIG metric for the objects tasked for observation, meaning it was rarely put in a position to receive an observation. However, the UKF and AEGIS-based FIG metrics for this

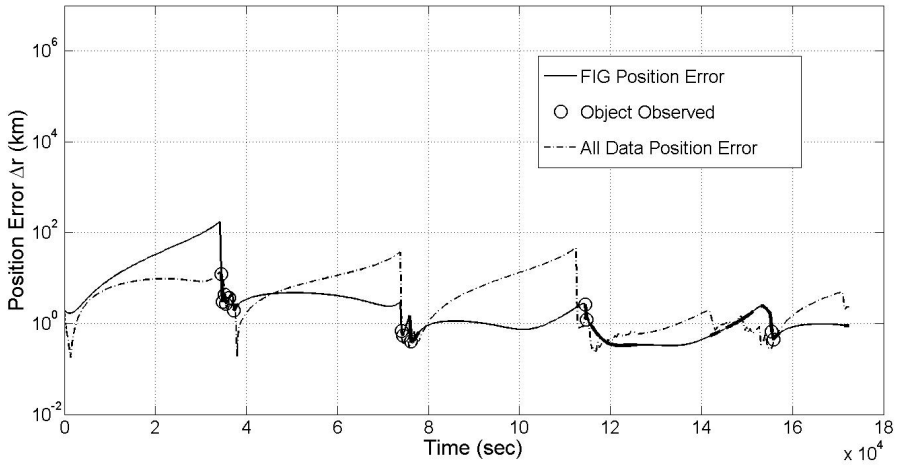

Figure 5.13. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 71 using FIG tasking strategy in conjunction with an AEGIS filter. Thick sections of line represent times when observations were available between object 71 and one or more sensors.

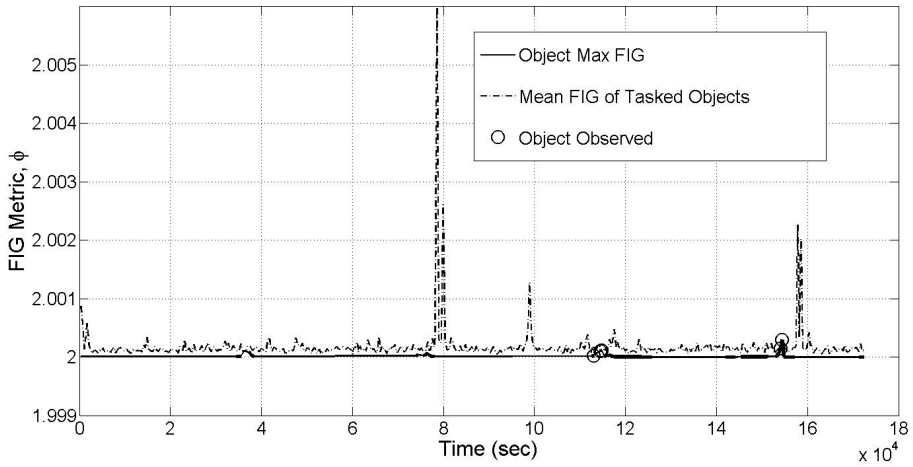

Figure 5.14. Plot of the FIG utility metric ϕ and tasking decisions for object 71 using FIG tasking strategy in conjunction with an EKF. Thick sections of line represent times when observations were available between that object and one or more sensors.

object managed to have many more observations, due to the fact that the FIG metric, while generally lower than the average, had several instances when it was competitive with respect to the FIG metrics of objects available for observation. Of additional note is that the EKF-based FIG value for object 71, as well as the average among all objects had fairly homogeneous time histories, where for the UKF and AEGIS filter many more spikes occurred. For object 71 these spikes

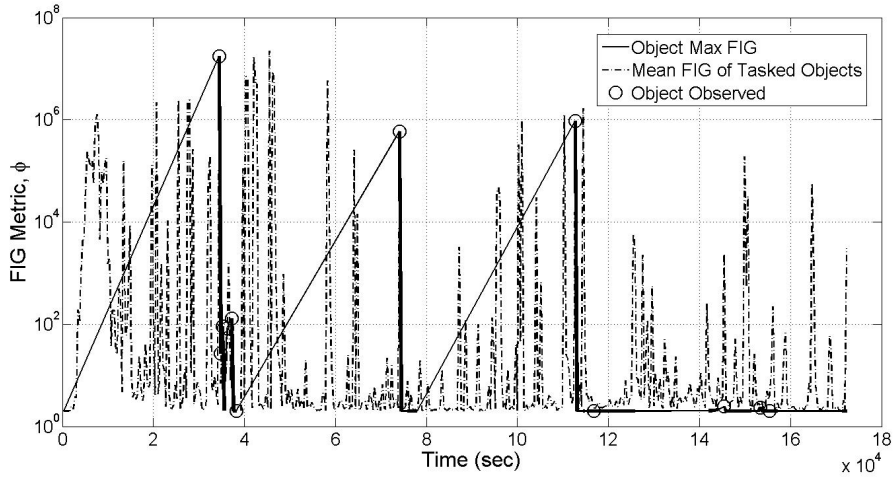

Figure 5.15. Plot of the FIG utility metric ϕ and tasking decisions for object 71 using FIG tasking strategy in conjunction with a UKF. Thick sections of line represent times when observations were available between that object and one or more sensors.

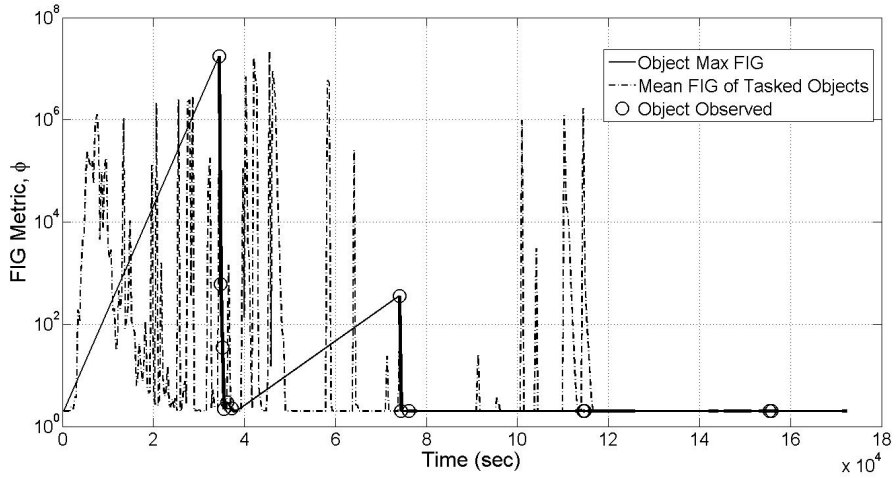

Figure 5.16. Plot of the FIG utility metric ϕ and tasking decisions for object 71 using FIG tasking strategy in conjunction with an AEGIS filter. Thick sections of line represent times when observations were available between that object and one or more sensors.

first occurred through approximately the first two thirds of the simulation, and then diminished. This behavior matches that of the UKF and AEGIS position error plot for object 71 in Figures 5.12 - 5.13, where errors would experience significant growth at times between updates, until approximately two thirds of the way through the simulation where they stabilized. However, this was not the case

for the FIG metric when used with an EKF, where the error and utility metric were completely uncorrelated. This phenomenon can be easily explained by specifically calculating the FIG metric for an EKF. By applying Eq. 4.25 to Eq. 4.24, the result is

$$\eta_{(i,j),k+1} = H_{(i,j),k+1} = \begin{bmatrix} \frac{\hat{x}^f_{i,k+1} - x^s_{j,k+1}}{\rho_{(i,j),k+1}} & \frac{\hat{y}^f_{i,k+1} - y^s_{j,k+1}}{\rho_{(i,j),k+1}} & 0 & 0 \\ -\frac{\hat{y}^f_{i,k+1} - y^s_{j,k+1}}{\rho^2_{(i,j),k+1}} & \frac{\hat{x}^f_{i,k+1} - x^s_{j,k+1}}{\rho^2_{(i,j),k+1}} & 0 & 0 \end{bmatrix} \tag{5.10}$$

which if applied to Eq. 4.34 results in a metric whose calculation depends only on the objects prediction step estimate location with respect to the sensor location, and the sensor noise covariance (which for the low-error simulations is constant for each sensor). Furthermore, when evaluating the trace of the FIG matrix using an EKF, this will only contain information on position, since the diagonal elements of the FIG matrix associated with velocity information will always be zero. This illustrates a direct consequence the EKF linearizations (in this case, the linearization of the nonlinear measurement function) have on the evaluation of the FIG metric, and results in the poorly distributed EKF tasking decisions in Figures 5.10 and 5.11, as well as FIG metric behavior seen in Figure 5.14. The only way to alleviate this consequence would be to ensure that EKF measurement equations contain functions of all the state elements to avoid the zero diagonal values in the FIG matrix calculation.

Additionally, the scale of the FIG utility metric values vary widely between the EKF and other filters, another consequence of the zero-valued velocity information diagonals in the EKF-FIG matrix. Since velocity variances for this simulation can be on the order of 10^{-6}, it is easy to see how information metrics (which roughly incorporate the inverse of these variances) could produce large discrepancies between the values of FIG utility metrics for the EKF and UKF/AEGIS filters as seen in Figures 5.14 - 5.16.

If similar steps are taken for a UKF or AEGIS filter, the value for $\eta_{(i,j),k+1}$ will not only have non-zero diagonal elements of the FIG matrix, but they will also be functions of the distribution of sigma points. In particular, both the distribution

of the sigma points in the prediction step, and their transformation into measurement space will factor into the calculation of $\eta_{(i,j),k+1}$ and therefore $\phi_{(i,j),k+1}$. This results in the FIG metric having a less homogeneous time-history for the UKF and AEGIS filter than the EKF, adding more diversity to the elements in the visibility matrix, which increases the chances for more evenly distributed tasking decisions and overall better performance.

From observing Figures 5.15 - 5.16, the behavior is very similar, except at the beginning of the simulation where the AEGIS filter provided much more updates, and towards the end of the simulation when average FIG values for the AEGIS filter began to stabilize and produced less updates than the UKF. Though object 71 ended on roughly the same estimated position error for both the UKF and AEGIS filter, the average estimated position error over the entire simulation was lower when using the AEGIS filter than the UKF, as shown in Figure 5.9. Figure 5.13 shows that this better average estimated position error was probably due to the simulation time span from approximately 40,000 - 110,000 seconds, where the AEGIS filter produced more updates early in the simulation, resulting in less divergence in estimation error during segments when no updates occurred. This shows that while the UKF and AEGIS filter had approximately the same amount of updates for this object, they occurred at different times in the simulation. The fact that these occurred earlier in the simulation for the AEGIS filter, leading to less error (almost one order of magnitude less) than the UKF for a significant portion of the simulation illustrates the slight benefits of using the AEGIS filter to provide FIG tasking schedules over the UKF.

5.3.3 SIG Tasking

Table 5.7 shows the scalar performance metrics outlined in Eqs. 5.5 and 5.8 for the low-error simulation using SIG tasking. In addition, figures showing the distribution of position errors and how these errors compared to the estimated $3\hat{\sigma}^r_{i,k}$ provided by the filter covariance estimates are presented in Figures 5.17 - 5.18. Results show that each filter obtains better performance than in the implementation of the FIG tasking strategy, particularly the EKF. Unlike in the FIG simulation, the SIG tasking strategy does not have the disproportionately bad performance for the EKF, as shown in the comparable average error metric (with respect to the UKF and AEGIS filter) in Table 5.6, along with a position estimate errors ($\Delta r_{i,k}$) greater than 25 km in Figure 5.17, and percentage of position estimate errors greater than $3\hat{\sigma}^r_{i,k}$ in Figure 5.18. While the EKF still produced the most position errors outside the $3\hat{\sigma}^r_{i,k}$ bounds in Figure 5.18 (only slightly higher than the AEGIS filter), it also produced estimated position errors in a distribution very similar (only slightly worse) to the UKF.

For the UKF-SIG combination, the UKF produced very little position errors outside of the $3\hat{\sigma}^r_{i,k}$ bounds as shown in Figure 5.18, and managed to obtain better position errors as shown by Table 5.7 and Figure 5.17 than in the FIG simulation.

The AEGIS filter once again produced the best performance using the SIG tasking, as reflected by having the lowest scalar performance metrics in Table 5.7. Furthermore, Figure 5.7 shows that the AEGIS filter provided a disproportionate amount of estimates within the tightest position error bounds of $\Delta r_{i,k} \leq 1$ km, as it did in the all data and FIG simulations. When observing Figure 5.8, while the AEGIS filter produced more position errors outside of the $3\hat{\sigma}^r_{i,k}$ bounds than the UKF, these estimates were few (less than 1%). What is more important is that the AEGIS filter produced the most position errors within $0.8\hat{\sigma}^r_{i,k} \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}^r_{i,k}$, meaning that while it produced slightly more position errors outside the $3\hat{\sigma}^r_{i,k}$ bounds than the UKF, it also generally produced more accurate uncertainty estimates than the UKF or EKF.

Regarding overall performance, the SIG tasking produced better estimates for

all filters than in the FIG simulation, with the reduction in scalar performance metrics in Table 5.7, and increase in position errors within $\Delta r_{i,k} \leq 1$ km for all filters.

Table 5.7. Simulation Performance Metrics for SIG Low-Error Simulation

Filter	$\langle E_r^* \rangle$ (km)	$\left\langle \hat{A}_{err} \right\rangle$ (km^2)	$\frac{\langle E_r^* \rangle_{Tasking}}{\langle E_r^* \rangle_{All\ Data}}$	$\frac{\langle \hat{A}_{err} \rangle_{Tasking}}{\langle \hat{A}_{err} \rangle_{All\ Data}}$
EKF	7.361	3.938×10^2	1.335	1.085
UKF	7.206	3.057×10^2	1.307	1.102
AEGIS	5.789	2.744×10^2	1.250	1.068

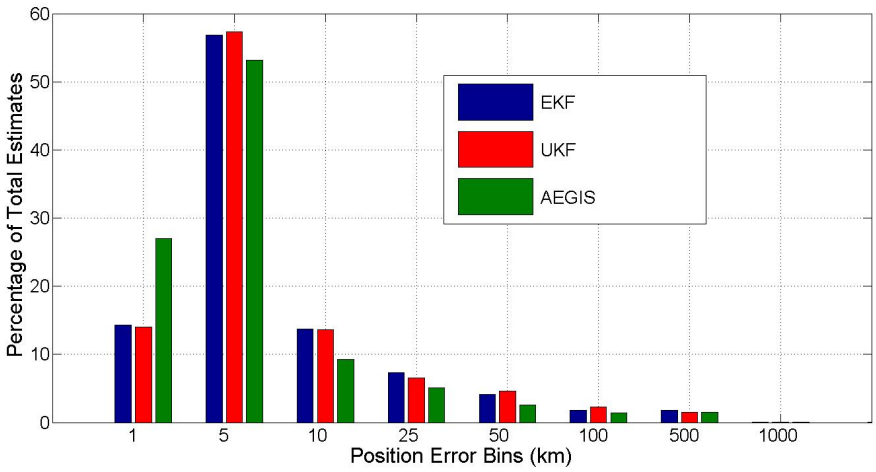

Figure 5.17. EKF, UKF, and AEGIS histograms of position estimate error distributions for SIG simulation. Plots include data from all objects at all simulation time steps.

To investigate why performance discrepancies existed in the implementation of EKF, UKF, and AEGIS filters in conjunction with an SIG tasking strategy, Figures 5.19 - 5.20 show the amount of updates each simulated object received along with their corresponding average position errors in the SIG simulation.

As shown in Figure 5.20, when using SIG tasking there was a more even distribution of observations as opposed to the FIG simulation, with the AEGIS filter heavily favoring certain objects for observation (most notably objects 21, 34, 35 and 39), and the UKF and EKF providing a slightly more even distribution of updates (except for objects 49 and 57). The result is that fewer objects accumu-

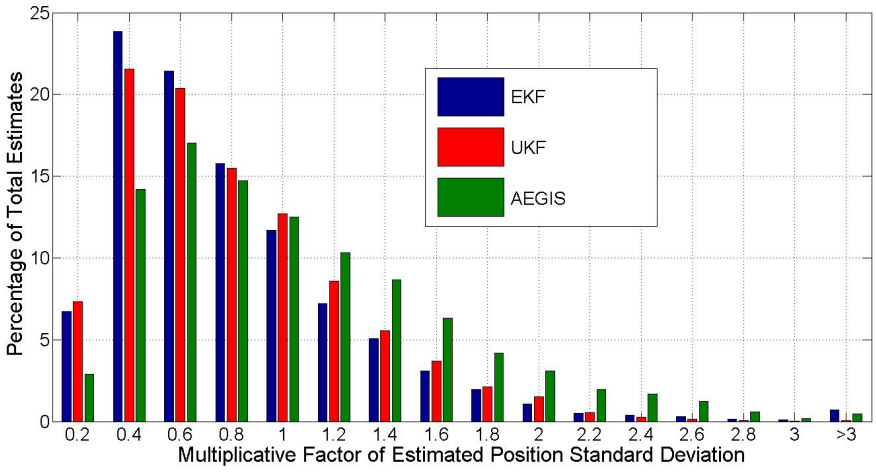

Figure 5.18. EKF, UKF, and AEGIS histograms of $J_{i,k}^{\hat{P}}$ metric in Eq. 5.9 for SIG simulation. Plots include data from all objects at all simulation time steps

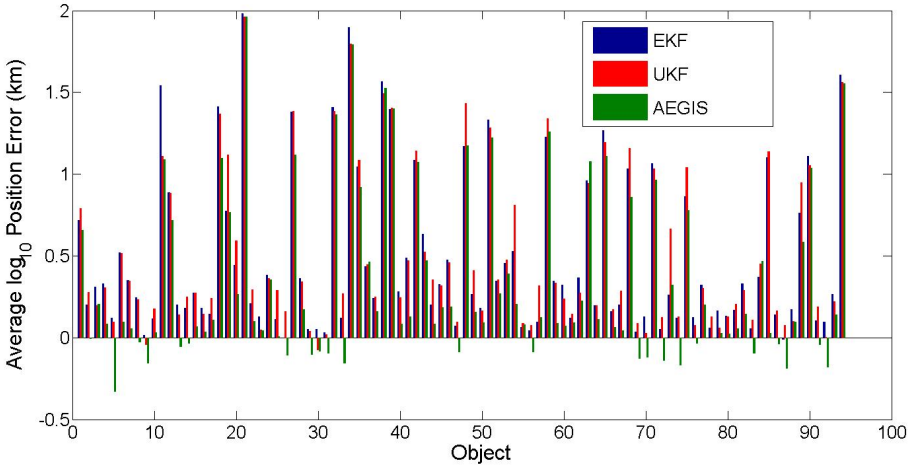

Figure 5.19. Bar graphs showing the $\log_{10}$ average position error for each object given in Eq. 5.7, through implementation of an EKF, UKF and AEGIS filter in conjunction with an SIG tasking strategy.

lated large estimate errors than what was experienced using FIG tasking, and in particular many objects that received little to no observations in the FIG simulation received several observations in the SIG simulation. This had a drastic effect on the performance of the EKF relative to the UKF and AEGIS filter in the SIG simulation as compared to the FIG simulation, as shown in Figure 5.19.

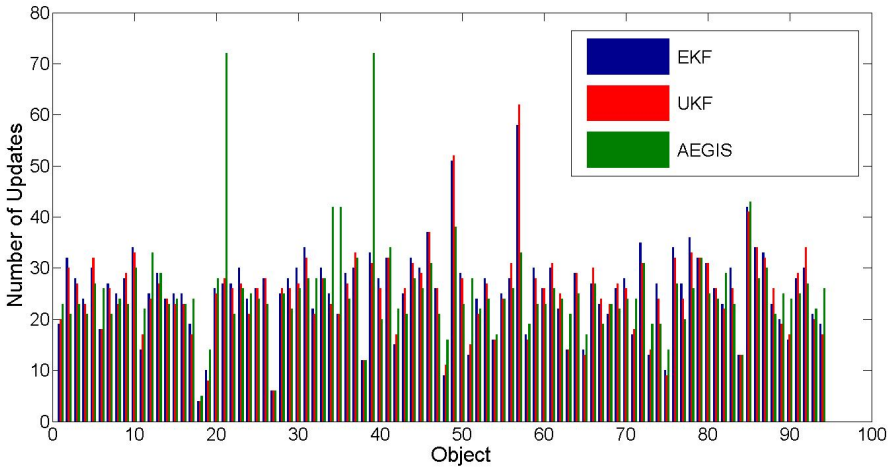

Figure 5.20. Bar graphs showing the updates each object received through implementation of an EKF, UKF, and AEGIS filter in conjunction with an SIG tasking strategy.

While the UKF provided a slight overall performance advantage to the EKF, Figure 5.19 shows that the EKF produced lower average position estimate errors for several objects. The difference was that for objects which accumulated the largest errors, the EKF would have a larger error than the UKF or AEGIS filter (objects 11, 21, 34, 94), which would have resulted in a higher overall average estimated position error in Table 5.7. Furthermore, Figure 5.19 shows that the UKF and in particular the AEGIS filter produced lower average estimated position errors for the best performing objects than the EKF (e.g. objects 5, 9, 30, etc).

The EKF, UKF and AEGIS filters still maintained differences in observations between all objects, for which the large amount of observations the AEGIS-SIG combination produced for several objects (e.g. objects 21, 34, 35, 39) was the most obvious difference. For these objects where the AEGIS produced many more observations than the EKF or UKF, the AEGIS filter was able to produce equal or better average estimated position errors. However, it was also the case that for several objects which received relatively low updates using the AEGIS filter, average estimated position errors were still lower than when using the EKF or UKF (e.g. objects 2, 49, and 86). This points to the AEGIS filter providing efficient sensor schedules so that it could achieve better performance using less estimates for many objects, occasionally leading to an excessive amount of updates for some

objects which accumulated the largest errors (e.g. objects 21 and 34). Also, in a similar manner to the FIG simulation, the AEGIS filter would generally provide equal or more updates to objects which were available for few observations (e.g. objects 18, 19, 42, etc), and fewer observations for objects which were often available (most notably object 57). This makes intuitive sense that objects with fewer availabilities for observations should be given preference over objects which are readily available, and points to the AEGIS filter being more efficient than either the EKF or UKF in terms of the sensor schedules which resulted from its implementation with SIG.

To investigate why sensor tasking differed between the EKF, UKF, and AEGIS filters when implementing SIG, Figures 5.21 - 5.26 show when object 21 (the worst performing SIG object using either filter) was available for observation, tasked for observation, as well as the time histories of its position estimate error using SIG, the position error in the all data case, its maximum SIG utility metric (at times it was available for observation), and the average SIG metric for objects tasked for observation. Figures 5.24 - 5.26 show that object 21 was not even available for observation until a simulation time of about 110,000 seconds, where each filter provided immediate tasking observations. The major differences between the filters in this case was that the UKF and AEGIS filter provided much better updates (resulting in lower position errors) than the EKF from a time span of 110,000 - 130,000 seconds, and that the AEGIS filter provided much more observations (more than twice as many) than the EKF or UKF.

In Figures 5.24 - 5.26, the SIG metric for object 21 varies little between the EKF and UKF filters, while there is a noticeable difference using the AEGIS filter. The first difference is that the mean SIG for tasked objects consistently decreases throughout the simulation, where it stays roughly the same for the EKF and UKF. The second is that the SIG metric for object 21 remains closer to the mean for the AEGIS filter than the EKF or UKF, resulting in more observations using the AEGIS filter. In either case, it is obvious that there is something which is driving the AEGIS filter to produce different values for SIG than the EKF or UKF, resulting in different tasking schedules, and overall better performance.

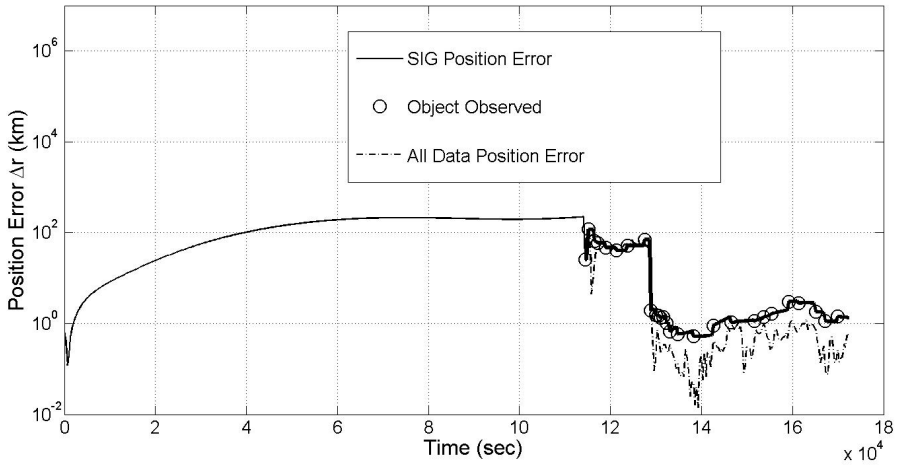

Figure 5.21. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 21 using SIG tasking strategy in conjunction with an EKF. Thick sections of line reflect times when observations were available between object 21 and one or more sensors.

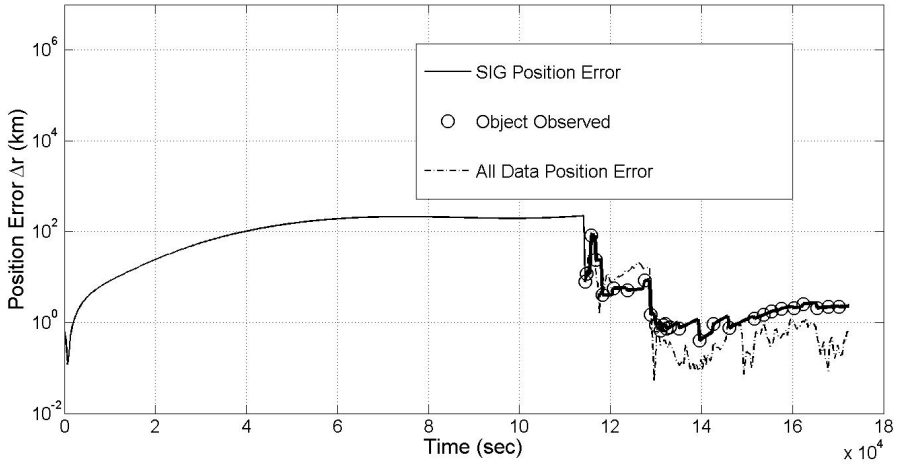

Figure 5.22. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 21 using SIG tasking strategy in conjunction with a UKF. Thick sections of line reflect times when observations were available between object 21 and one or more sensors.

To investigate this, Figure 5.27 shows the evolution of the estimated error ellipse area for the EKF, UKF, and AEGIS filter for object 21. This figure is chosen since the SIG metric uses determinants of the covariance estimate in its calculation, and since the calculation of $\hat{A}_{err}$ takes the square root of the product of

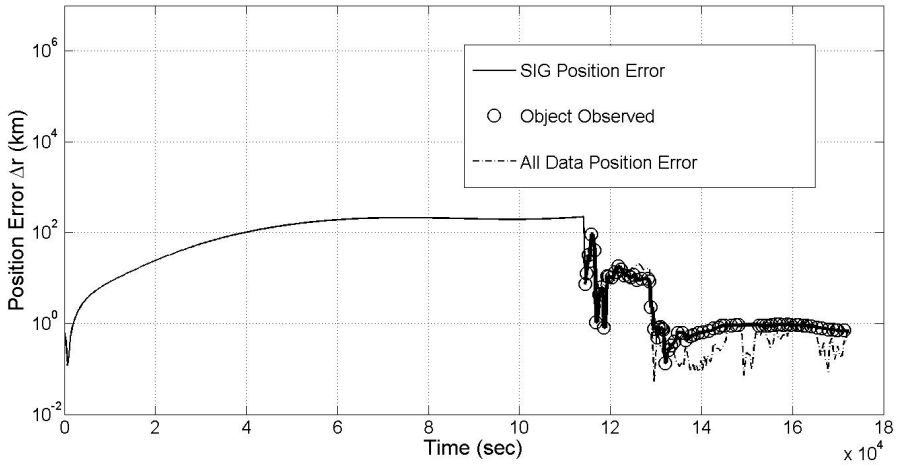

Figure 5.23. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 21 using SIG tasking strategy in conjunction with an AEGIS Filter. Thick sections of line reflect times when observations were available between object 21 and one or more sensors.

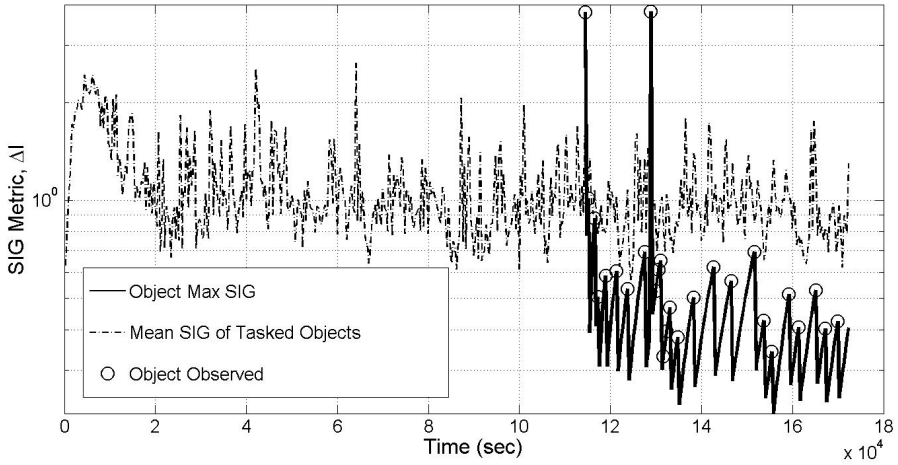

Figure 5.24. Plot of the SIG utility metric ΔI and tasking decisions for object 21 using SIG tasking strategy in conjunction with an EKF. Thick sections of line reflect times when observations were available between that object and one or more sensors.

position covariance eigenvalues[2], the determinant of the covariance and estimated error ellipse area will be strongly correlated. Therefore, from Figure 5.27, it is obvious that the time histories of $\hat{A}_{err}$ differ between the filters, as they did in the all data simulation illustrated in Figure 5.4. The primary difference is that the

[2]Recall the product of eigenvalues is the same as the determinant of a matrix

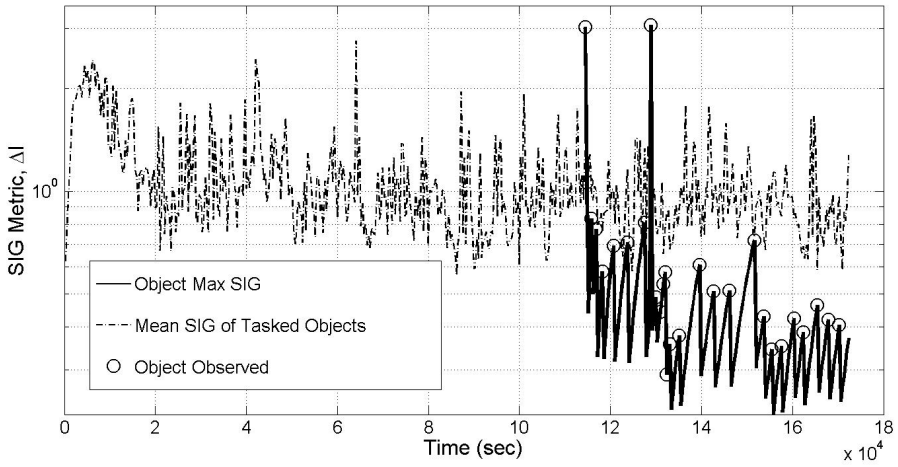

Figure 5.25. Plot of the SIG utility metric ΔI and tasking decisions for object 21 using SIG tasking strategy in conjunction with a UKF. Thick sections of line reflect times when observations were available between that object and one or more sensors.

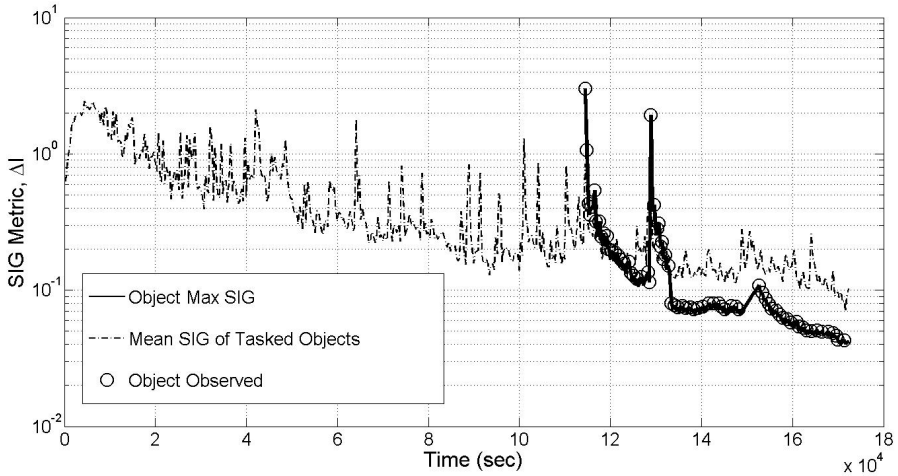

Figure 5.26. Plot of the SIG utility metric ΔI and tasking decisions for object 21 using SIG tasking strategy in conjunction with an AEGIS filter. Thick sections of line reflect times when observations were available between that object and one or more sensors.

AEGIS filter typically gets much better covariance updates resulting in a lower value of $\hat{A}_{err}$ when updates occur, and therefore less divergence in $\hat{A}_{err}$ in subsequent time steps when updates do not occur. Since the SIG metric relies on the ratio of determinants of covariance estimates between the forecast and update steps, if this change is greater for the AEGIS filter than other filters, it will result in a different evolution in the SIG metric as well. In the case of object 21 using an

AEGIS filter, the result is that the SIG metric decreases more over time, due to less divergence in $\hat{A}_{err}$ during the forecast step, and less reduction in $\hat{A}_{err}$ during observations (since there is less potential to obtain information for covariance estimates which are lower with respect to the EKF and UKF). Furthermore, if the AEGIS filter provides a larger reduction in covariance estimates during observations than the EKF or UKF, it will take less simulation time for many objects to obtain low covariance estimates, and therefore objects with larger uncertainty (such as object 21, due to the long period of no observations in the first half of the simulation) should be allocated more observations, such as the case with object 21 (and the other objects with the worst errors as shown in Figures 5.19 - 5.20).

These findings could be extended when comparing the EKF and UKF, in that the same rationale applies regarding the propagation and updating of the covariance estimate. In this case, the UKF provides on average slightly better updates than the EKF, and can provide better propagation depending on the nonlinearity of the governing dynamics. In this case, there is a noticeable difference in the propagation of $\hat{A}_{err}$ (and therefore related to a difference in the propagation of covariance estimates) from the start of the simulation until about 110,000 seconds. In addition, the first series of updates just after 110,000 seconds results in a lower value of $\hat{A}_{err}$ for the UKF than the EKF, indicating the UKF received greater reduction in the covariance estimate than the EKF at that time. However, since these better updates resulted in a lower value of $\hat{A}_{err}$ for only a small portion of the simulation, the differences in sensor tasking between the EKF and UKF for object 21 were slight, with the UKF only receiving one more additional update than the EKF, while the AEGIS filter received 45 additional updates.

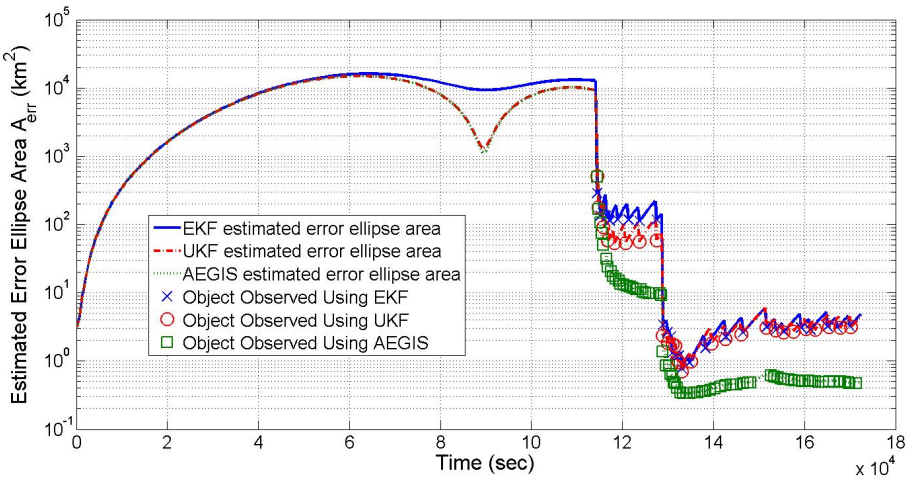

Figure 5.27. EKF, UKF, and AEGIS plots of $\hat{A}_{err} = \pi\sqrt{\hat{\varsigma}^1_{i,k}\hat{\varsigma}^2_{i,k}}$ metric for object 21 in the SIG simulation.

5.3.4 LLE Tasking

Table 5.8 shows the scalar performance metrics outlined in Eqs. 5.5 and 5.8 for the low-error simulation using LLE tasking. In addition, figures showing the distribution of position errors and how these errors compared to the estimated $3\hat{\sigma}^r_{i,k}$ provided by the filter covariance estimates are presented in Figures 5.28 - 5.29.

Table 5.8. Simulation Performance Metrics for LLE Low-Error Simulation

Filter	$\langle E^*_r \rangle$ (km)	$\left\langle \hat{A}_{err} \right\rangle$ (km^2)	$\frac{\langle E^*_r \rangle_{Tasking}}{\langle E^*_r \rangle_{All\ Data}}$	$\frac{\left\langle \hat{A}_{err} \right\rangle_{Tasking}}{\left\langle \hat{A}_{err} \right\rangle_{All\ Data}}$
EKF	7.999	4.018×10^2	1.451	1.107
UKF	7.687	3.095×10^2	1.394	1.116
AEGIS	6.324	2.751×10^2	1.365	1.070

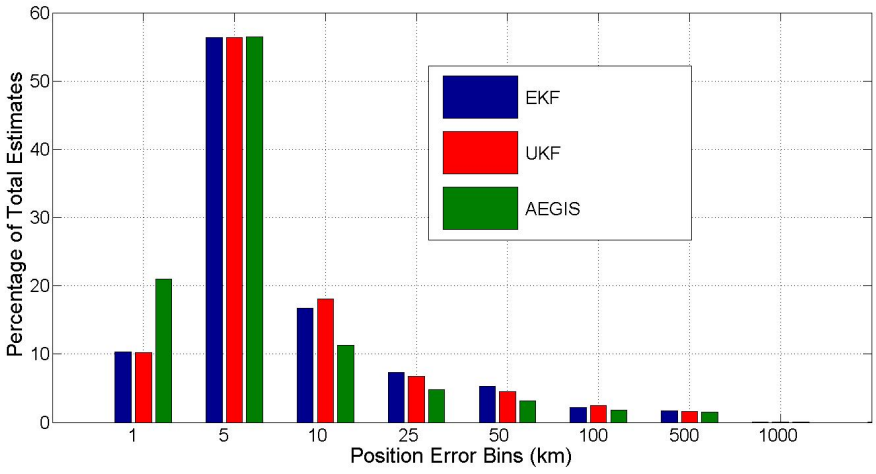

Figure 5.28. EKF, UKF, and AEGIS histograms of position estimate error distributions for the LLE simulation. Plots include data from all objects at all simulation time steps.

When observing the performance utilizing LLE tasking, it provides better estimates for all filters than the FIG strategy, and worse than the SIG method. In addition, the EKF and UKF/AEGIS filter do not show the same magnitude of performance discrepancies as they did in the case of using FIG-based tasking. From observing Figures 5.28 - 5.29, using an LLE tasking strategy results in more objects within position estimate errors of 5 km, and much less (especially for the EKF) greater than 100 km, or greater than $3\hat{\sigma}^r_{i,k}$ when compared to the FIG strategy. However, when compared to the SIG results, it does not provide better results in any of these categories. In addition, just as the case with FIG and SIG tasking

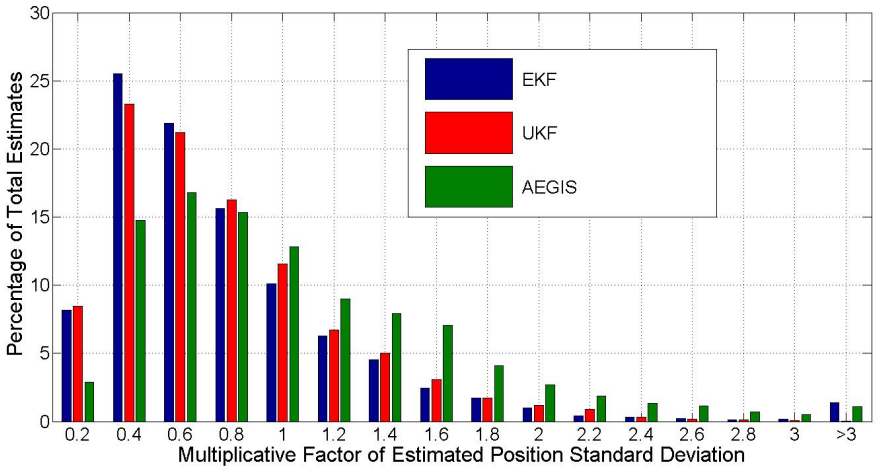

Figure 5.29. EKF, UKF, and AEGIS histograms of $J^{\hat{P}}_{i,k}$ metric in Eq. 5.9 for the LLE simulation. Plots include data from all objects at all simulation time steps

strategies, performance discrepancies did exist between the application of EKF, UKF, and AEGIS filters, as shown by the ratio of LLE average position error compared to the all data case as shown in the third column of Table 5.8. In this case, the EKF provided the worst performance, while the AEGIS filter provided the best. The reason for the EKF's poor performance with respect to the other filters was that it produced the least percentage of position estimates within 10 km and within $0.8\hat{\sigma}^r_{i,k} \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}^r_{i,k}$, the most outside of 10 km and the greater than $3\hat{\sigma}^r_{i,k}$.

For the UKF-LLE combination, the UKF performed the best in terms of producing few position errors outside of the $3\hat{\sigma}^r_{i,k}$ bounds as shown in Figure 5.29, and managed to obtain better position errors as shown by Table 5.8 and Figure 5.28 than in the FIG simulation, but still did not generate the quality of estimates as in the SIG simulation. Furthermore, the UKF produced the second most percentage of errors within 10 km and $0.8\hat{\sigma}^r_{i,k} \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}^r_{i,k}$, falling short only to the AEGIS filter.

The AEGIS filter once again produced the best performance using the LLE tasking, as reflected by having the lowest scalar performance metrics in Table 5.8. Furthermore, Figure 5.28 shows that the AEGIS filter provided a disproportionate

amount of estimates within the tightest position error bounds of $\Delta r_{i,k} \leq 1$ km, as it did in the all data, FIG, and SIG simulations. When observing Figure 5.8, while the AEGIS filter produced more position errors outside of the $3\hat{\sigma}^r_{i,k}$ bounds than the UKF, these estimates were few (less than 2%). What is more important is that the AEGIS filter produced the most position errors within $0.8\hat{\sigma}^r_{i,k} \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}^r_{i,k}$, meaning that while it produced slightly more position errors outside the $3\hat{\sigma}^r_{i,k}$ bounds than the UKF, it also generally produced more accurate uncertainty estimates than the UKF or EKF.

To investigate why performance discrepancies existed in the implementation of the EKF, UKF, and AEGIS filters in conjunction with an LLE tasking strategy, Figures 5.30 - 5.31 show the amount of updates each simulated object received along with their corresponding average position errors in the LLE simulation.

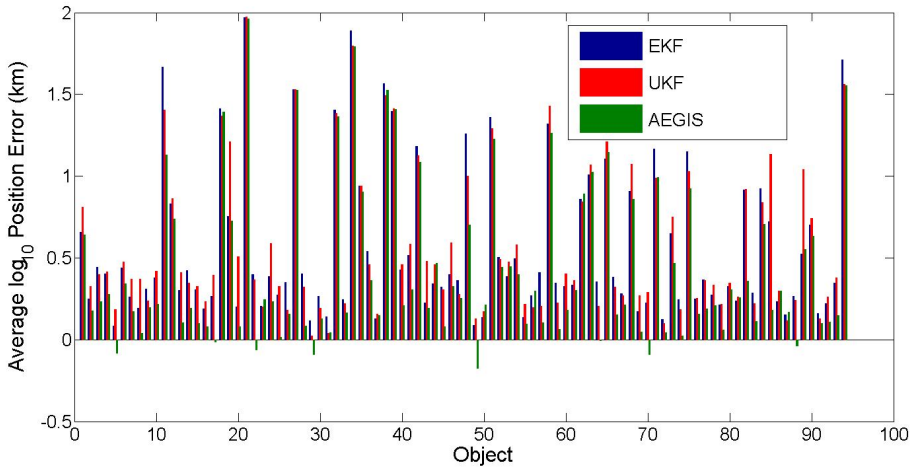

Figure 5.30. Bar graphs showing the $\log_{10}$ average position error for each object given in Eq. 5.7, through implementation of an EKF, UKF and AEGIS filter in conjunction with an LLE tasking strategy.

In contrast to Figure 5.10, Figure 5.31 shows a more even distribution of observations among all the simulated objects in much the same way as Figure 5.20. Furthermore, while several objects have a minor difference in the amount of updates received using an EKF, UKF, or AEGIS filter, these differences are nowhere near the magnitude shown in Figure 5.10, except for a few objects using the AEGIS

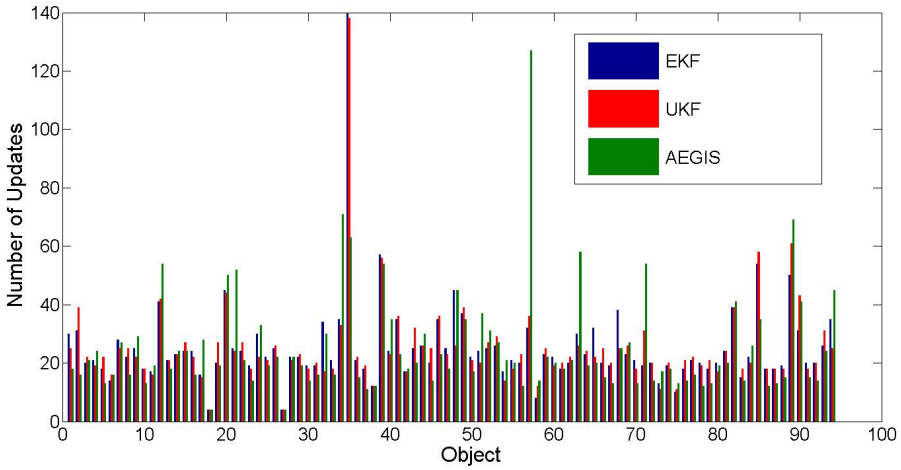

Figure 5.31. Bar graphs showing the updates each object received through implementation of an EKF, UKF, and AEGIS filter in conjunction with an LLE tasking strategy.

filter (e.g. objects 57, 63, and 71). What is interesting though is that for most of the objects with the worst position estimate errors (i.e. objects 11, 21, 27, 34, 38, 94, etc.) the EKF either had an approximately equal or moderately worse average position estimate error than the UKF or AEGIS filters, though these objects often received more updates than the UKF, and much less than the AEGIS filter. In fact, just as the case with using SIG tasking, the worst performing objects received a disproportionate amount of updates using the AEGIS filter. These differences in tasking decisions show the UKF can provide less observations to a poor performing object than the EKF while still maintaining better estimates than the EKF, while the AEGIS filter is able to allocate a large amounts of sensor resources to a few poorly performing objects while still maintaining typically better performance for other objects as compared to the EKF or UKF. In fact, many objects which were able to obtain the best performance under the AEGIS filter were given the least amounts of updates with respect to the EKF and UKF, such as objects 5, 22, 49, 70, and 88.

To examine the LLE metric, and how it effects tasking decisions between the EKF, UKF, and AEGIS filter, Figures 5.32 - 5.37 show when object 21 (the worst performing LLE object using either filter) was available for observation, tasked for observation, as well as the time histories of its position estimate error using LLE,

the position error in the all data case, its maximum LLE utility metric (at times it was available for observation), and the average LLE metric for objects tasked for observation.

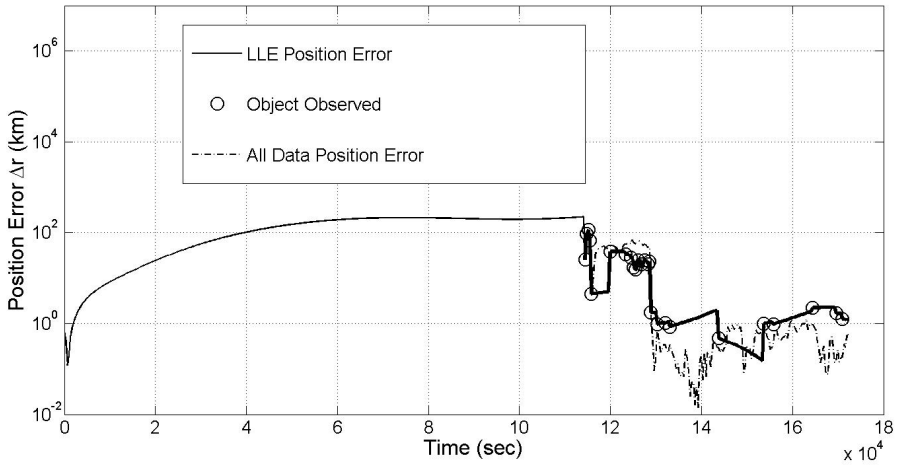

Figure 5.32. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 21 using the LLE tasking strategy in conjunction with an EKF. Thick sections of line reflect times when observations were available between object 21 and one or more sensors.

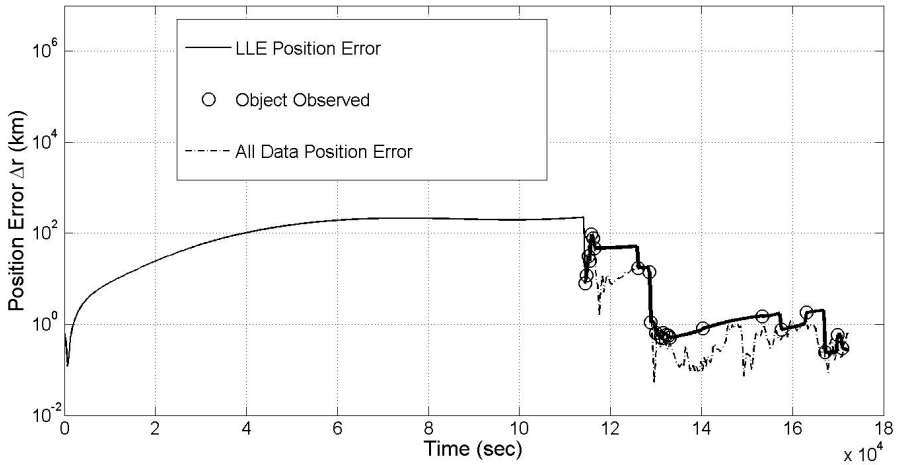

Figure 5.33. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 21 using the LLE tasking strategy in conjunction with a UKF. Thick sections of line reflect times when observations were available between object 21 and one or more sensors.

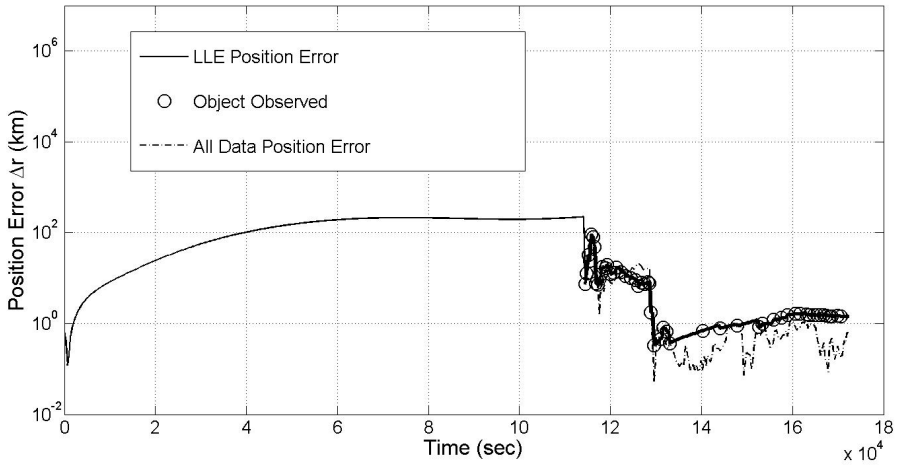

Figure 5.34. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 21 using the LLE tasking strategy in conjunction with an AEGIS filter. Thick sections of line reflect times when observations were available between object 21 and one or more sensors.

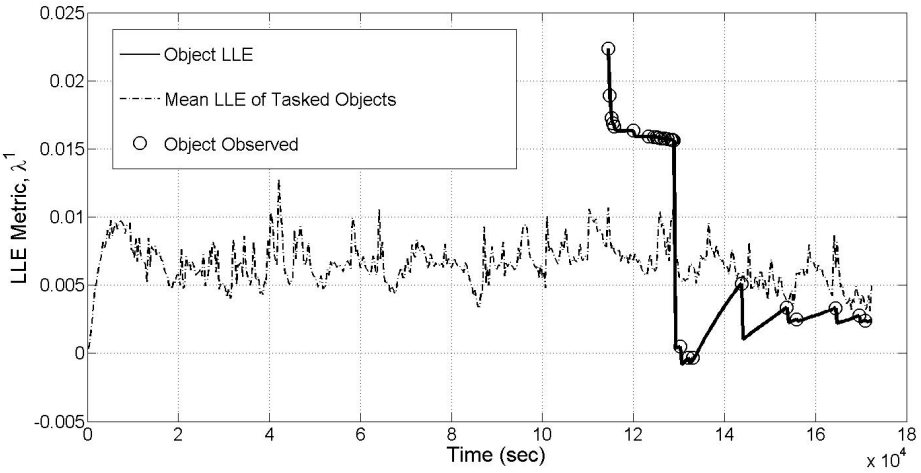

Figure 5.35. Plot of the LLE utility metric $\hat{\lambda}^1$ and tasking decisions for object 21 using the LLE tasking strategy in conjunction with an EKF. Thick sections of line reflect times when observations were available between that object and one or more sensors.

Figures 5.32 - 5.33 show that the worst object in the LLE simulation was one that experienced a very long time span when observations were not possible, between times of approximately 0 - 110,000 seconds (this is the same object with the worst performance in the all data and SIG simulations). For all filters, the object was immediately observed at the first instance it was available, which was one trait the LLE metric was hypothesized to provide, as described in Chapter 4.

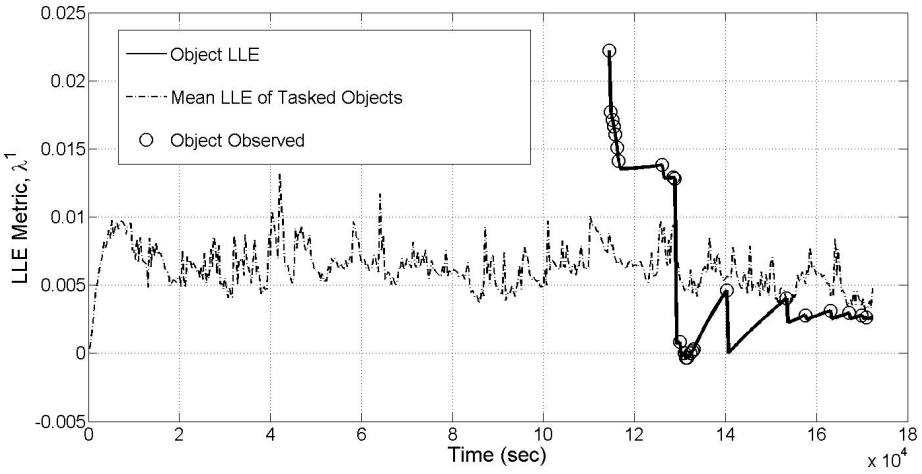

Figure 5.36. Plot of the LLE utility metric $\hat{\lambda}^1$ and tasking decisions for object 21 using the LLE tasking strategy in conjunction with a UKF. Thick sections of line reflect times when observations were available between that object and one or more sensors.

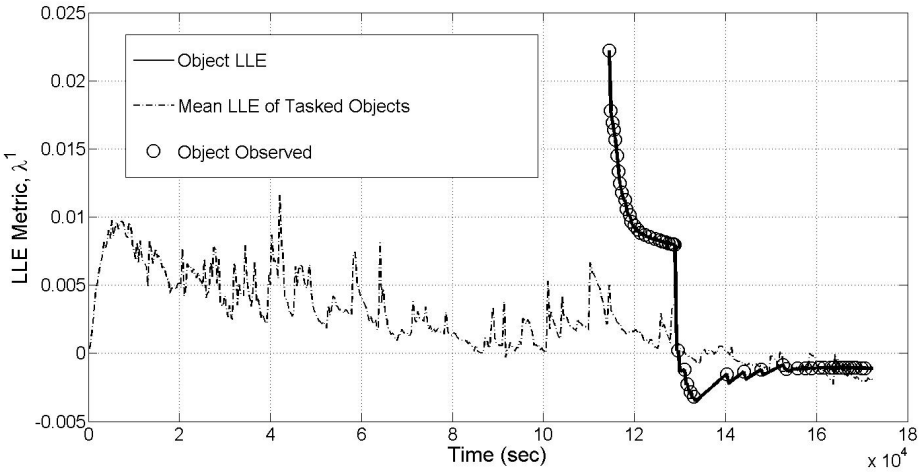

Figure 5.37. Plot of the LLE utility metric $\hat{\lambda}^1$ and tasking decisions for object 21 using the LLE tasking strategy in conjunction with an AEGIS filter. Thick sections of line reflect times when observations were available between that object and one or more sensors.

Since object 21 had similar numbers of updates between the EKF and UKF (25 for the EKF, 24 for the UKF), there was little difference between performance, though the UKF ended with a better position estimate than the EKF. This slight difference of one observation follows the trend that for the objects with the worst estimation errors, the UKF resulted in slightly less observations than the EKF, as

was observed in Figure 5.31. Figure 5.34 differs from Figures 5.32 - 5.33 primarily in how many more updates object 21 received (52 observations) using the AEGIS filter as opposed to the EKF or UKF. This was more than double what object 21 received for either the EKF or UKF, and follows the same trait that was observed using SIG tasking in that the AEGIS filter provides a disproportionate amount of updates to the worst performing objects. What is interesting is that it does this without sacrificing performance on the objects which receive less updates, indicating that in the case of SIG or LLE tasking, the AEGIS filter provides the most efficient tasking decisions.

Figures 5.35 - 5.37 show that the LLE metric has a strong correlation to the actual estimate error as the simulation progresses (i.e. rises and falls with estimate error). The result is that the LLE metric provides an effective prioritization scheme which gives preference to observe objects with diverging estimation errors, resulting in the few objects with errors greater than 100 km as seen in Figure 5.30.

In addition, as was the case for the SIG simulation, the LLE metric for the AEGIS filter behaves slightly different than the EKF or UKF, in that its average value of tasked objects decreases more over time than for the EKF or UKF. As explained in the SIG simulation, this is the result of the advantages in obtaining the covariance estimate using the AEGIS filter as opposed to the EKF or UKF. While the LLE and SIG metrics differ in their calculation, they both require a scalar quantification of the estimated covariance matrix in their calculation. In the case of the LLE metric, this is the square root of the trace of the forecast step covariance, which is normalized with respect to the square root of the trace of the initial covariance estimate (equal for all objects). Therefore, for the LLE metric to consistently decrease in its time history (in both the overall average and of object 21), it implies that the covariance estimates are continuously decreasing using the AEGIS filter, where for the EKF and UKF they are somewhat stabilized. This fact that covariance estimates experience a greater reduction per observation using the AEGIS filter as opposed to the EKF or UKF also explains why the worst objects are given so many observations using the AEGIS filter because the average LLE value among all objects decreases more using the AEGIS filter. As with the

SIG simulation, these observations point to the AEGIS filter providing not only an estimation advantage, but also a tasking advantage using the LLE strategy as opposed to the EKF or UKF.

5.4 High-Error Simulation

As described in Chapter 2, 100 objects are distributed with semi major axes and eccentricities shown in Figure 5.38. Even though 100 objects were originally created, as with in the low-error simulation, only 94 of these objects made passes within at least one sensor's field of regard during the simulation time span. Since there is no purpose in evaluating performance of objects which could never have been physically observed by any sensors, all simulation performance is calculated with respect to the $N_s = 94$ objects which had the possibility for observation.

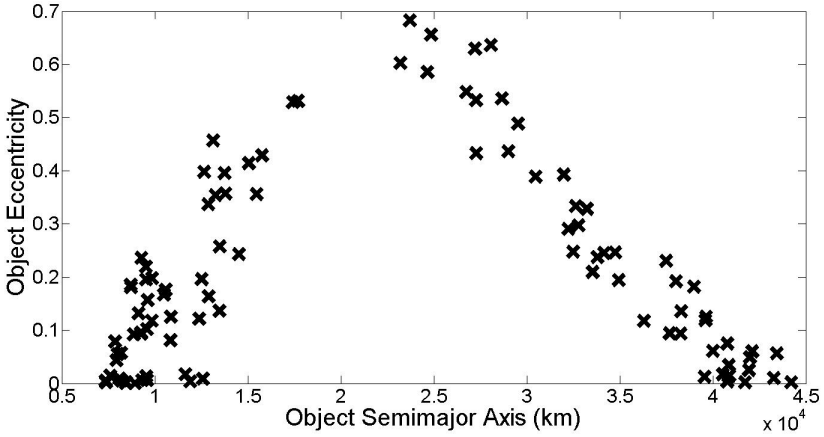

Figure 5.38. Distribution of semimajor axes and eccentricities of 100 objects in high-error simulation

Initial object state standard deviations (used in both selecting the initial state estimates, and defining the initial covariance estimate) for the high-error test case are given in Table 5.9.

Table 5.9. Initial Standard Deviations: High-Error Simulation

Standard Deviation	Value
$\hat{\sigma}^x_{i,0}$	10 (km) $\forall i$
$\hat{\sigma}^y_{i,0}$	10 (km) $\forall i$
$\hat{\sigma}^{\dot{x}}_{i,0}$	10^{-3} (km/sec) $\forall i$
$\hat{\sigma}^{\dot{y}}_{i,0}$	10^{-3} (km/sec) $\forall i$

Additionally, sensor noise in increased and given in Table 5.10. In this case, the noise for each sensor is a function of the effective range and half angle of that

sensor (i.e. noise increases as effective range increases).

Table 5.10. Sensor Noise for High-Error Simulations

Sensor (j)	v_j^ρ (km)	v_j^ψ (deg)
1	2.055	0.0134
2	4.318	0.0031
3	3.273	0.0044
4	1.1936	0.0071
5	3.559	0.0039

5.4.1 All Data Tasking

Histograms showing percentage of position errors falling within the bins described in Section 5.2 for the all data simulation is presented in Figure 5.39. Another histogram showing the percentage of position errors falling within specified multiplicative factors of the estimated position standard deviation are found in Figure 5.40. Additionally, tabulated results showing the scalar performance metrics outlined in Eqs. 5.5 and 5.8 are presented in Table 5.11.

Table 5.11. Simulation Performance Metrics for All Data High-Error Simulation

Filter	$\langle E_r^* \rangle$ (km)	$\left\langle \hat{A}_{err} \right\rangle$ (km^2)
EKF	18.442	9.516×10^2
UKF	15.071	8.465×10^2
AEGIS	13.850	7.853×10^2

From observing Figures 5.39 - 5.40 as well as Table 5.11, the AEGIS filter once again has the best performance in the all data simulation, followed by the UKF and EKF, as was the case in the low-error simulation. While scalar performance metrics were higher than in the low-error case, many of the trends from the low-error simulation continued into the high-error simulation. One trend that was not repeated was the difference in scalar performance metrics between the EKF and UKF filters. In the high-error case, the inferior initial state/covariance estimates and updates of these estimates (due to the higher sensor noise) have a more adverse effect on the implementation of the EKF than the UKF or AEGIS filters.

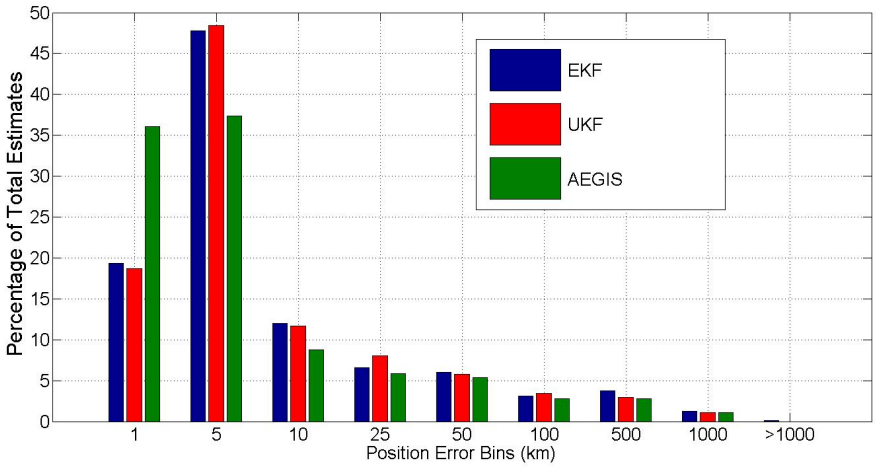

Figure 5.39. EKF, UKF, and AEGIS histograms of position estimate error distributions for all data simulation. Plots include data from all objects at all simulation time steps.

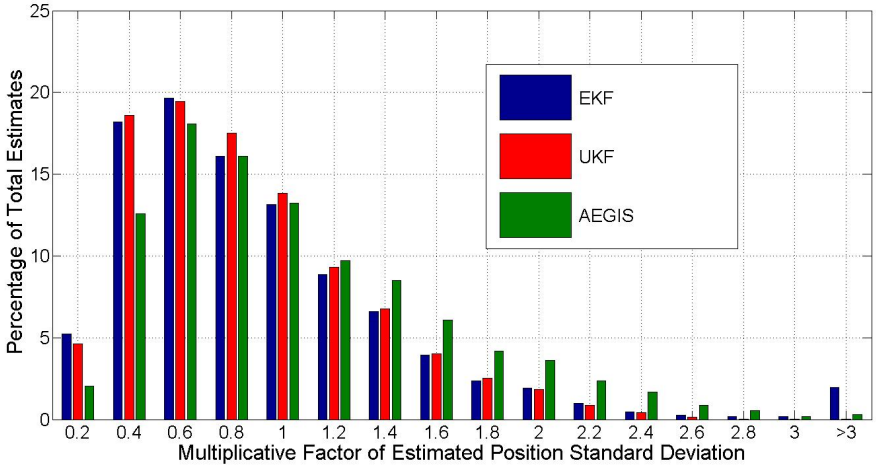

Figure 5.40. EKF, UKF, and AEGIS histograms of $J_{i,k}^{\hat{P}}$ metric in Eq. 5.9 for all data simulation. Plots include data from all objects at all simulation time steps.

This results in the EKF producing a worse average position error as seen in Table 5.11 (where in the low-error simulation the EKF and UKF produced roughly equal values), and a small percentage of position errors $\Delta r_{i,k} \geq 1000$ km (opposed to none in the low-error case) or increase of position estimates $\Delta r_{i,k} \geq 3\hat{\sigma}_{i,k}^{r}$ as seen in Figures 5.39 - 5.40 respectively.

The AEGIS filter once again performs the best due to the fact that it had the

lowest average position error (resulting from the greatest percentage of position estimates within 1 km), as well as the lowest average estimated error ellipse area. Figure 5.40 also shows that while the AEGIS filter provides the lowest average estimated error ellipse area, it still occasionally has overconfident uncertainty estimates (as shown by having a small percentage of position errors $\Delta r_{i,k} \geq 3\hat{\sigma}^r_{i,k}$). Additionally, just as in the low-error simulation, the UKF and the AEGIS filter produced the most position errors within $0.8\hat{\sigma}^r_{i,k} \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}^r_{i,k}$, as shown in Figure 5.40. However, in this case the two produced roughly an equal percentage of estimates in this range, unlike in the low-error simulation where the AEGIS filter produced more.

To investigate the propagation of the hyper volume of uncertainty (more accurately, a metric which is proportional to that hyper volume), Figure 5.41 shows the propagation of the average error ellipse area in Eq. 5.8 for the object with the worst average position error. Much like in the low-error simulation, the AEGIS filter consistently provides the lowest error ellipse area, indicating a lower extent of estimate uncertainty, which is validated by the lower average position estimate errors obtained by the AEGIS filter, as shown in the Table 5.11 and Figure 5.39. The major difference between the the same results for the low-error case (Figure 5.4) is that in many time periods, the UKF actually has a higher average error ellipse area than the EKF. This is the result of the EKF's tendency to be overconfident in its estimates, as described previously. The low value for $\hat{A}_{err}$ for the EKF as compared to the UKF for this simulation coincides with a greater tendency to produce inaccurate error estimates (as illustrated by the large percentage of EKF position estimates $\Delta r_{i,k} \geq 3\hat{\sigma}^r_{i,k}$), and therefore points to a fault and not an advantage with these lower EKF $\hat{A}_{err}$ values.

To investigate how each individual object performed in the all data simulation, Figure 5.42 shows the average position error for each simulated object, while Figure 5.43 shows how many updates each of these objects received. Figure 5.43 illustrates the diversity in the number of updates each object received, which Figure 5.42 shows is strongly correlated to the average position error an object would obtain throughout the simulation. Just as with the low-error simulation, some

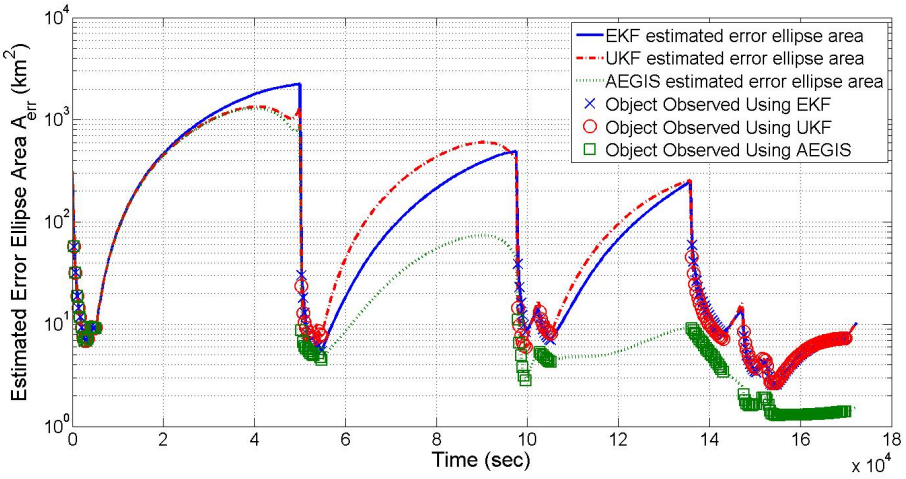

Figure 5.41. EKF, UKF, and AEGIS plots of $\hat{A}_{err} = \pi\sqrt{\hat{\varsigma}^1_{i,k}\hat{\varsigma}^2_{i,k}}$ metric for the object with the highest average position error in the all data simulation.

objects are sparsely available for observation (e.g. objects 1, 8, 76, among others), while other objects have several (most notably object 71). For all objects, the AEGIS filter had the greatest tendency to provide the best average position errors as opposed to the EKF or UKF, which both had a series of objects with the worst errors. In fact, of many of the objects which had the worst average errors, the EKF always performed worse than the UKF (e.g. objects 39, 54, 81, and 94).

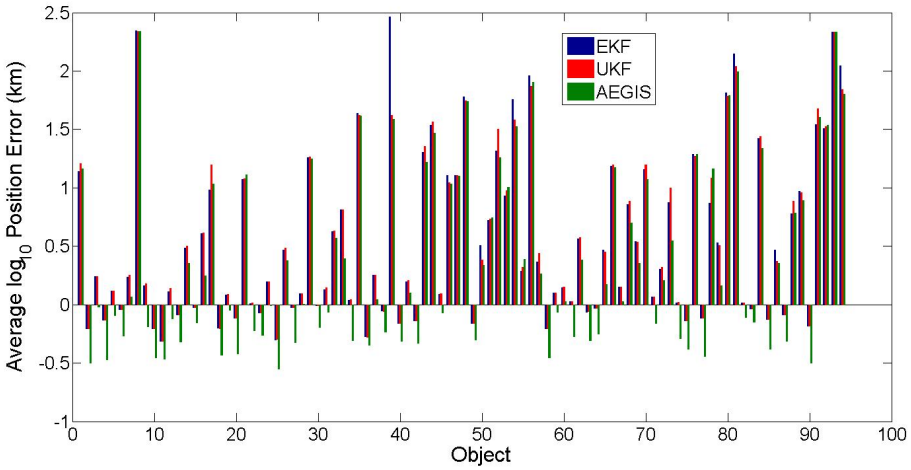

Figure 5.42. Bar graphs showing the $\log_{10}$ average position error for each object given in Eq. 5.7, through implementation of an EKF, UKF, and AEGIS filter in conjunction with all data tasking strategy. In this case, the filters have equal amounts of updates, reflecting the maximum updates possible for all objects.

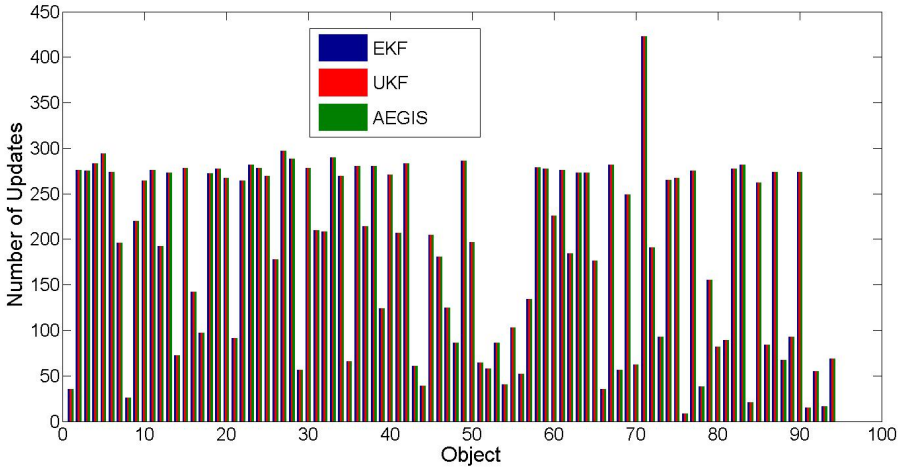

Figure 5.43. Bar graphs showing the updates each object received through implementation of an EKF, UKF, and AEGIS filter in conjunction with all data tasking strategy. In this case, the filters have equal amounts of updates, reflecting the maximum updates possible for all objects.

5.4.2 FIG Tasking

Table 5.12 shows the scalar performance metrics outlined in Eqs. 5.5 and 5.8 for the low-error simulation using FIG-based tasking. In addition, figures showing the distribution of position errors and how these errors compared to the estimated $3\hat{\sigma}^r_{i,k}$ provided by the filter covariance estimates are presented in Figures 5.44 - 5.45.

The EKF, UKF, and AEGIS filter all had similar performance using FIG-based tasking in the high-error test case as in the low-error, with the AEGIS filter performing the best, and the EKF experiencing a disproportionate drop in performance with respect to the other filters. This is shown in the very large average error metric in Table 5.12, along with a drastic increase of position estimate errors ($\Delta r_{i,k}$) greater than 100 km in Figure 5.44, as well as percentage of position estimate errors greater than $3\hat{\sigma}^r_{i,k}$ in Figure 5.45. In the latter case, the percentage of position estimate errors greater than $3\hat{\sigma}^r_{i,k}$ increased drastically in the high-error test case for the EKF, representing 12% of all estimates as opposed to 6% for the low-error FIG simulation. In contrast, the UKF and AEGIS filters did not experience this increase in poor estimates, indicating that the increase in errors led to much more drastic performance descrepancies when tasking was implemented than in the low-error simulations. This is also reflected in the relatively close proximity of the values for $\left\langle \hat{A}_{err} \right\rangle$ between the two filters in comparison to their large deviations in $\langle E^*_r \rangle$. This again implies that the EKF has a greater tendency to underestimate the possible error in its estimations than the UKF or AEGIS filters. The poor performance shown in the EKF-FIG combination would lead to either severely inaccurate object uncertainty and/or location estimates, or an inability to continue to monitor those objects should these methods be used in a real satellite tracking application.

As for the UKF-FIG combination, the performance was much better, and trends were similar to those in the low-error test case. Furthermore, the AEGIS filter also performed similar to the low-error test case, reflected by having the lowest scalar performance metrics in Table 5.12, the highest percentage of estimates within the tightest position error bounds of $\Delta r_{i,k} \leq 1$ km and the most position errors within $0.8\hat{\sigma}^r_{i,k} \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}^r_{i,k}$.

Table 5.12. Simulation Performance Metrics for FIG High-Error Simulation

Filter	$\langle E_r^* \rangle$ (km)	$\left\langle \hat{A}_{err} \right\rangle$ (km^2)	$\frac{\langle E_r^* \rangle_{Tasking}}{\langle E_r^* \rangle_{All\ Data}}$	$\frac{\left\langle \hat{A}_{err} \right\rangle_{Tasking}}{\left\langle \hat{A}_{err} \right\rangle_{All\ Data}}$
EKF	1.418×10^3	4.959×10^3	76.889	5.211
UKF	25.248	1.345×10^3	1.675	1.589
AEGIS	24.316	1.178×10^3	1.756	1.500

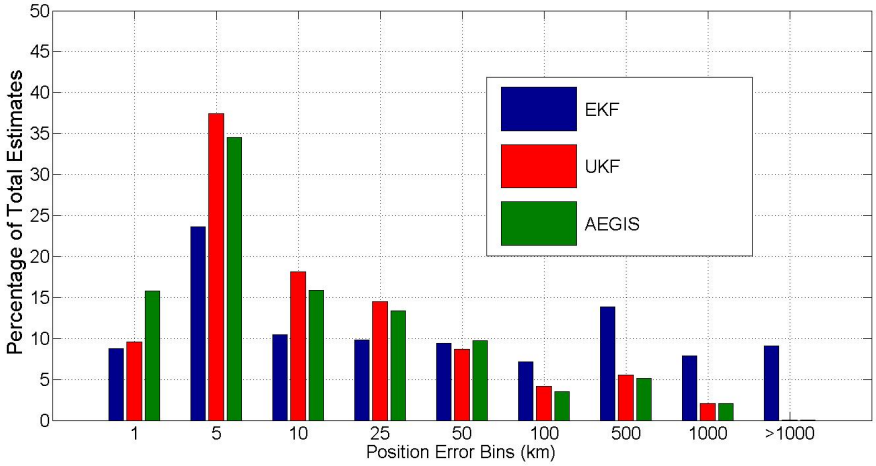

Figure 5.44. EKF, UKF, and AEGIS histograms of position estimate error distributions for FIG simulation. Plots include data from all objects at all simulation time steps.

To investigate why performance discrepancies existed in the implementation of EKF, UKF, and AEGIS filters in conjunction with an FIG tasking strategy, Figures 5.46 - 5.47 show the amount of updates each simulated object received along with their corresponding average position errors in the FIG simulation.

As shown in Figure 5.47, several objects received a disproportionately large amount of updates (e.g. objects 10, 11, 25, and especially 71) with respect to many others which received little to none. The result was generally the same as in the low-error test case, in that several objects accumulated large estimate errors due to these few updates, while those that had the most observations had very good average position estimate errors. In particular, the objects which received little to no observations had errors which were much greater when using an EKF as opposed to a UKF or AEGIS filter (e.g. objects 12, 24, 31, 32, etc.), which

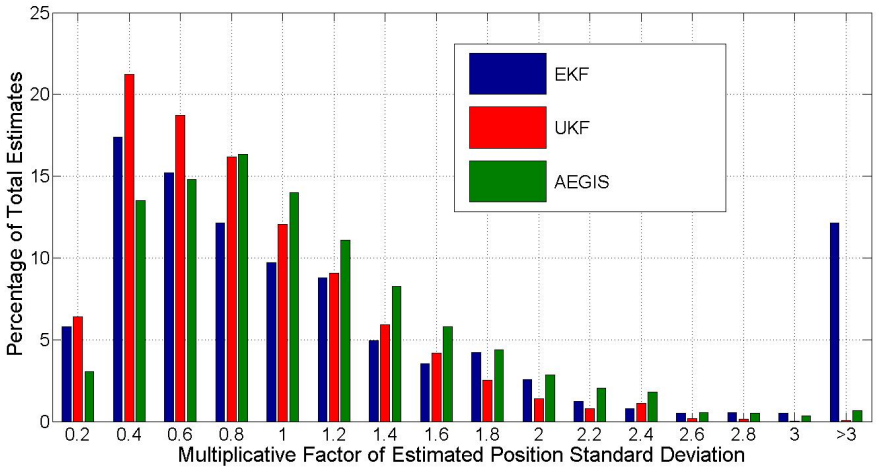

Figure 5.45. EKF, UKF, and AEGIS histograms of $J_{i,k}^{\hat{P}}$ metric in Eq. 5.9 for FIG simulation. Plots include data from all objects at all simulation time steps

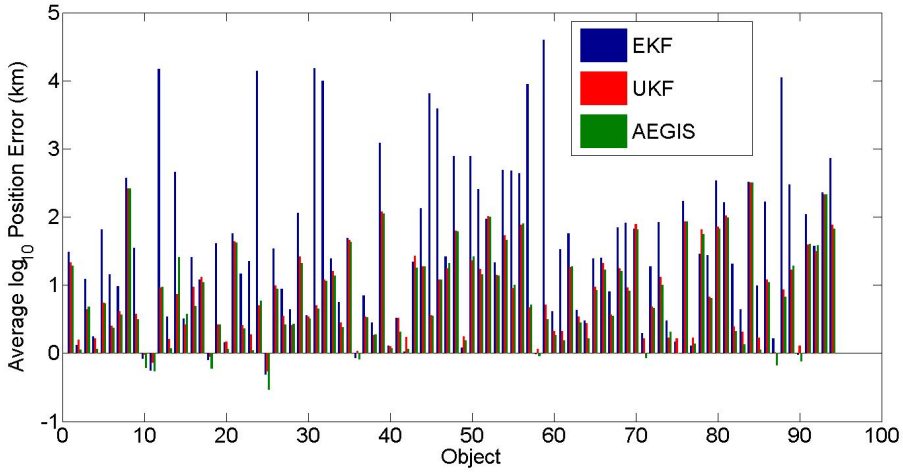

Figure 5.46. Bar graphs showing the $\log_{10}$ average position error for each object given in Eq. 5.7, through implementation of an EKF, UKF and AEGIS filter in conjunction with an FIG tasking strategy.

resulted in the vast performance discrepancies between the EKF and other two filters as seen in Table 5.12, as well as Figures 5.44 - 5.45. Furthermore, Figure 5.47 shows that for the particular objects that received little updates, the UKF and AEGIS filter consistently had more updates than the EKF (which sometimes had zero), with the UKF usually having the most. In all of these cases, the AEGIS filter generally provided better or of equal quality estimates than the EKF or

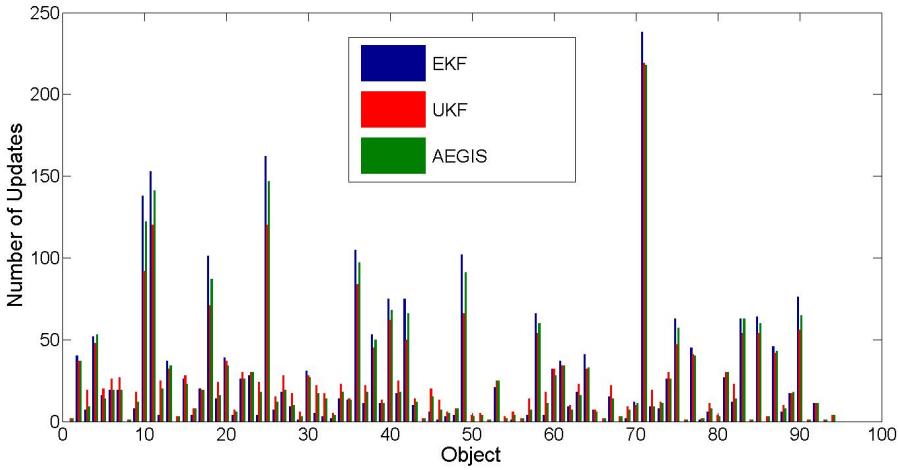

Figure 5.47. Bar graphs showing the updates each object received through implementation of an EKF, UKF, and AEGIS filter in conjunction with an FIG tasking strategy. As with the low-error scenario, the FIG tasking results in many updates to a select few objects, while many other objects receive little to none.

UKF, even at times when it resulted in less updates for a particular object than the UKF. Similarly, the EKF always produced more observations for those objects which had the lowest average position estimate errors, followed by the AEGIS filter and finally the UKF with the least updates (objects 10, 11, 18, 25, 36, etc.). This generally resulted in those objects having lower average position estimate errors using an EKF than a UKF, which the AEGIS errors would be lower than both.

To investigate why there was such a performance discrepancy between the EKF and other filters when implementing FIG, Figures 5.48 - 5.53 show when object 59 (the worst performing FIG object using either filter) was available for observation, tasked for observation, as well as the time histories of its position estimate error using FIG, the position error in the all data case, its maximum FIG utility metric (at times it was available for observation), and the average FIG metric for objects tasked for observation. Figure 5.48 shows that despite being available for observation for much of the simulation, using the EKF only resulted in 4 observations while the UKF resulted in 18 and the AEGIS filter 11. This increase in observations leads to the position error for this object to be much better, and much more stable for the UKF and AEGIS filter than the EKF where it diverged drastically.

In the case of the AEGIS filter, it was able to on average maintain a better position estimate error than the UKF throughout the whole simulation for object 59 using approximately two thirds of the updates.

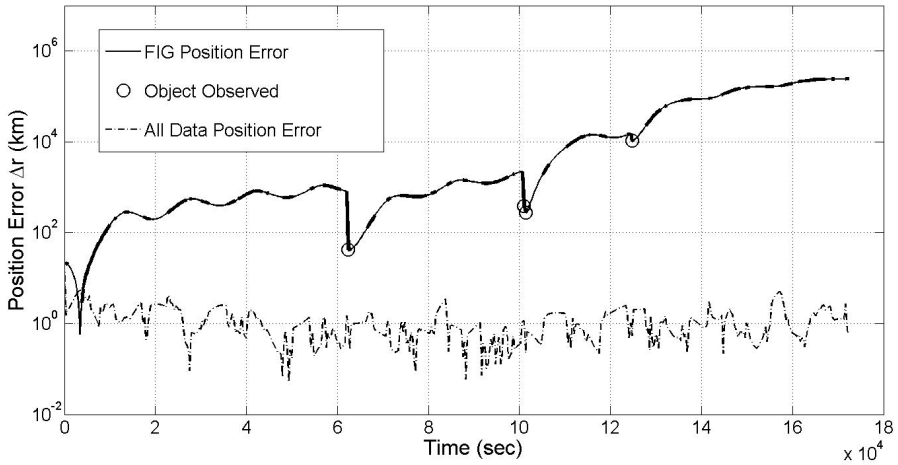

Figure 5.48. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 59 using FIG tasking strategy in conjunction with an EKF. Thick sections of line reflect times when observations were available between object 59 and one or more sensors.

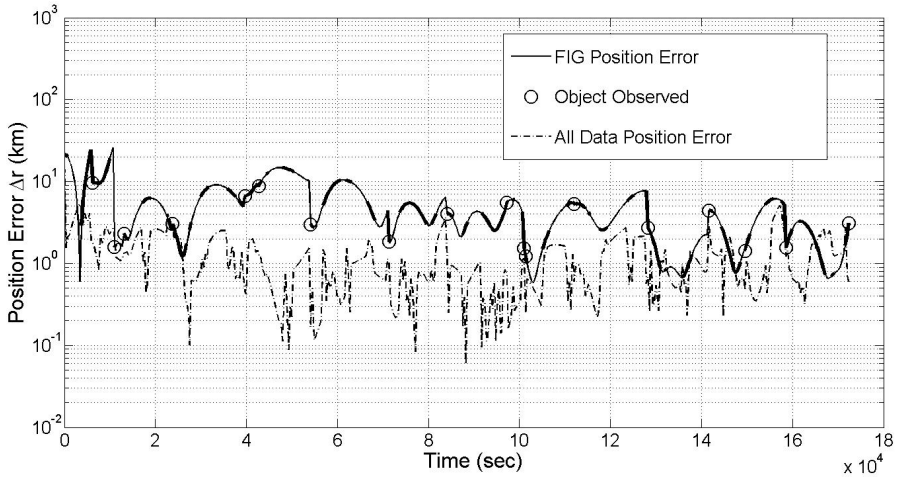

Figure 5.49. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 59 using FIG tasking strategy in conjunction with a UKF. Thick sections of line reflect times when observations were available between object 59 and one or more sensors.

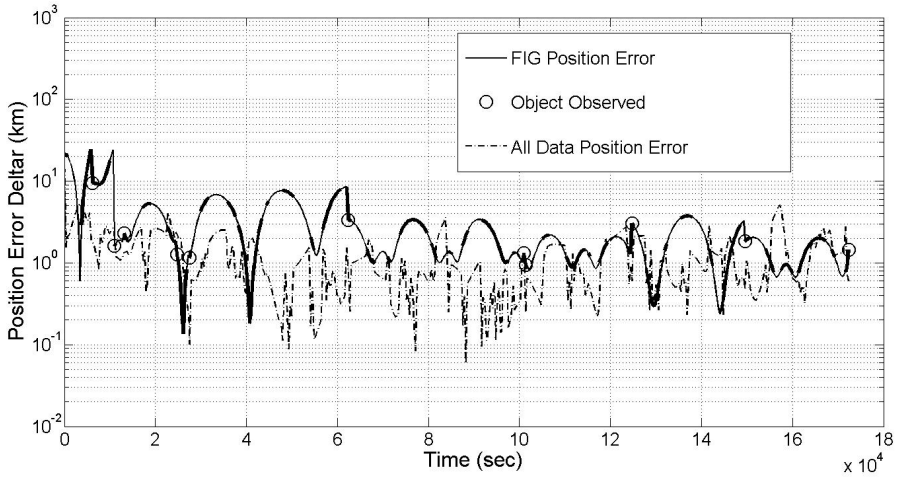

Figure 5.50. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 59 using FIG tasking strategy in conjunction with an AEGIS filter. Thick sections of line reflect times when observations were available between object 59 and one or more sensors.

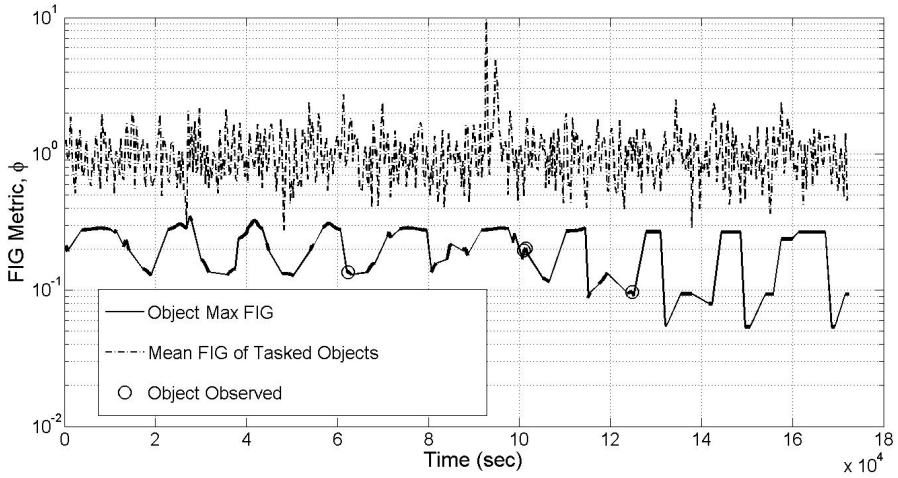

Figure 5.51. Plot of the FIG utility metric ϕ and tasking decisions for object 59 using FIG tasking strategy in conjunction with an EKF. Thick sections of line reflect times when observations were available between that object and one or more sensors.

When observing Figures 5.51 - 5.53, results were almost identical to the low-error test case. The EKF failed to produce updates to object 59 because its FIG metric was consistently below the average FIG for tasked objects, while the UKF and AEGIS filters produced FIG values more on par with the average of tasked objects. Furthermore, the FIG time history for the EKF was more homogeneous (less spikes) than for the UKF or AEGIS filter, due to the zero valued velocity

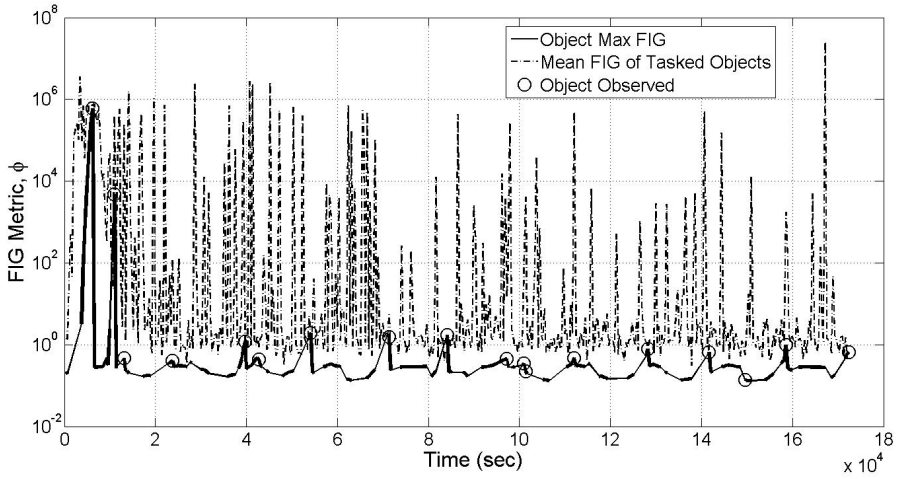

Figure 5.52. Plot of the FIG utility metric ϕ and tasking decisions for object 59 using FIG tasking strategy in conjunction with a UKF. Thick sections of line reflect times when observations were available between that object and one or more sensors.

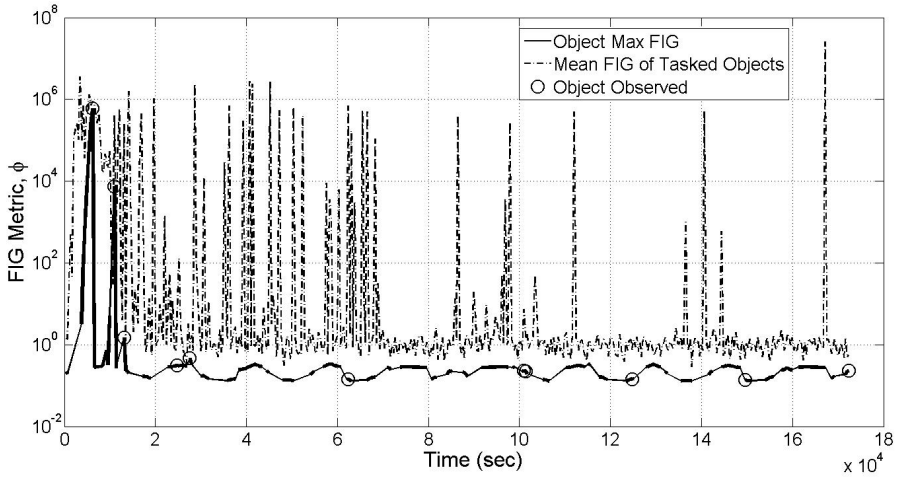

Figure 5.53. Plot of the FIG utility metric ϕ and tasking decisions for object 59 using FIG tasking strategy in conjunction with an AEGIS filter. Thick sections of line reflect times when observations were available between that object and one or more sensors.

information in the EKF's calculation of FIG, as detailed in Eq. 5.10. As with the low-error test case, the FIG metric for the EKF is uncorrelated with the filter's uncertainty (taking the form of sigma point distributions in the UKF and AEGIS filters) and therefore remains fairly constant throughout the simulation. The only differences between the high and low-error test cases were that there was slightly more variation in the FIG metric for the EKF (due to the sensor noise being dif-

ferent for each sensor), and that the spikes for object 59 in Figures 5.52 - 5.53 diminished earlier in the simulation. The latter is due to less divergence in the estimate errors for object 59 in the high-error case as opposed to object 71 in the low-error case, resulting in less possible information gain in the high-error case, and therefore less spikes in Figures 5.52 - 5.53 as opposed to Figures 5.15 - 5.16.

5.4.3 SIG Tasking

Table 5.13 shows the scalar performance metrics outlined in Eqs. 5.5 and 5.8 for the high-error simulation using SIG tasking. In addition, figures showing the distribution of position errors and how these errors compared to the estimated $3\hat{\sigma}^r_{i,k}$ provided by the filter covariance estimates are presented in Figures 5.54 - 5.55. Results show that each filter obtains better performance than in the implementation of the FIG tasking strategy in the same manner as the low-error test case. Also like in the low-error simulations, the SIG tasking strategy does not have the disproportionately bad performance for the EKF, though the EKF did not fare as well compared to the UKF and AEGIS filter as it did in the low-error simulation. This is shown in the relatively high average position error metric with respect to the UKF and AEGIS filter in Table 5.12, along with the small percentage of position estimate errors $\Delta r_{i,k} \geq 1000$ km in Figure 5.54, and percentage of position estimate errors greater than $3\hat{\sigma}^r_{i,k}$ in Figure 5.55. In the latter case, much like in the FIG high-error simulation, the amount of position estimates $\Delta r_{i,k} \geq 3\hat{\sigma}^r_{i,k}$ increased from the low-error SIG simulation for the EKF, while it remained roughly the same for the UKF and AEGIS filter. Once again, this points to the EKF being more suceptable to producing poor covariance (uncertainty) estimates as the amount of error increases in the simulation. This is compounded by the fact that the EKF also produced the most position errors $\Delta r_{i,k} \leq 0.2\hat{\sigma}^r_{i,k}$ and the least within $0.8\hat{\sigma}^r_{i,k} \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}^r_{i,k}$, meaning it produced the most overconfident (uncertainty estimate much smaller than actual error) and under confident (uncertainty estimate much larger than actual error) uncertainty estimates, while also producing the least accurate uncertainty estimates (uncertainty estimate is a fair approximation of actual error).

For the UKF-SIG combination, the UKF produced very little position errors outside of the $3\hat{\sigma}^r_{i,k}$ bounds as shown in Figure 5.55, and managed to obtain better position errors as shown by Table 5.13 and Figure 5.17 than in the FIG simulation. These results deviated vary little from the trends seen in the low-error simulation, except that the overall average errors and distribution of errors were of a higher magnitude.

The AEGIS filter once again produced the best performance using the SIG tasking as it did in the low-error simulations.. Furthermore, Figure 5.44 shows that the AEGIS filter provided a disproportionate amount of estimates within the tightest position error bounds of $\Delta r_{i,k} \leq 1$ km, as it did in the low-error SIG simulations. When observing Figure 5.45, while the AEGIS filter once again produced more position errors outside of the $3\hat{\sigma}^r_{i,k}$ bounds than the UKF, these estimates were few (less than 1%). What is more important is that the AEGIS filter produced the most position errors within $0.8\hat{\sigma}^r_{i,k} \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}^r_{i,k}$. Overall, like the UKF, there was little deviation in the trends observed for the AEGIS filter between the low and high-error simulations.

Table 5.13. Simulation Performance Metrics for SIG High-Error Simulation

Filter	$\langle E^*_r \rangle$ (km)	$\left\langle \hat{A}_{err} \right\rangle$ (km^2)	$\frac{\langle E^*_r \rangle_{Tasking}}{\langle E^*_r \rangle_{All\ Data}}$	$\frac{\left\langle \hat{A}_{err} \right\rangle_{Tasking}}{\left\langle \hat{A}_{err} \right\rangle_{All\ Data}}$
EKF	37.334	1.144×10^3	2.024	1.202
UKF	18.642	1.060×10^3	1.237	1.589
AEGIS	16.370	9.485×10^2	1.182	1.208

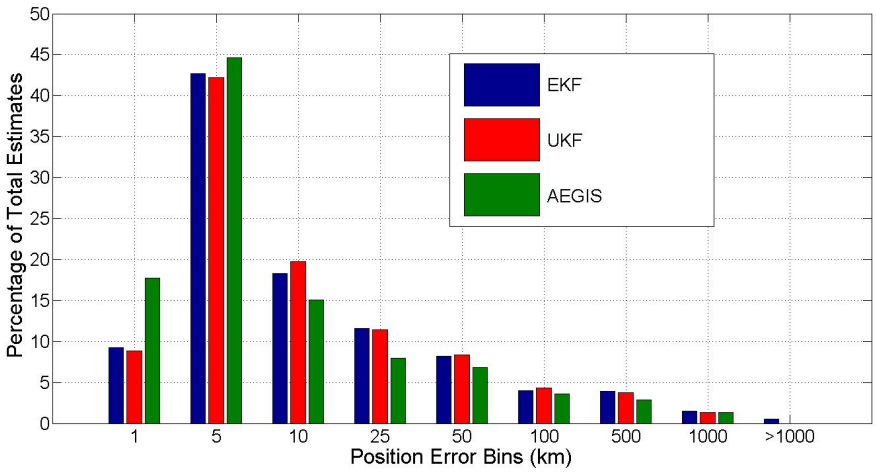

Figure 5.54. EKF, UKF, and AEGIS histograms of position estimate error distributions for SIG simulation. Plots include data from all objects at all simulation time steps.

To investigate why performance discrepancies existed in the implementation of EKF, UKF, and AEGIS filters in conjunction with an SIG tasking strategy, Figures 5.56 - 5.57 show the amount of updates each simulated object received along

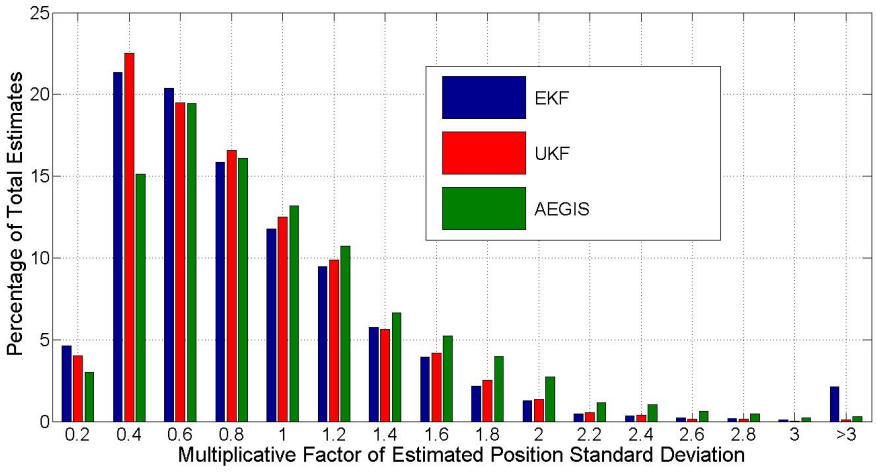

Figure 5.55. EKF, UKF, and AEGIS histograms of $J_{i,k}^{\hat{P}}$ metric in Eq. 5.9 for SIG simulation. Plots include data from all objects at all simulation time steps.

with their corresponding average position errors in the SIG simulation.

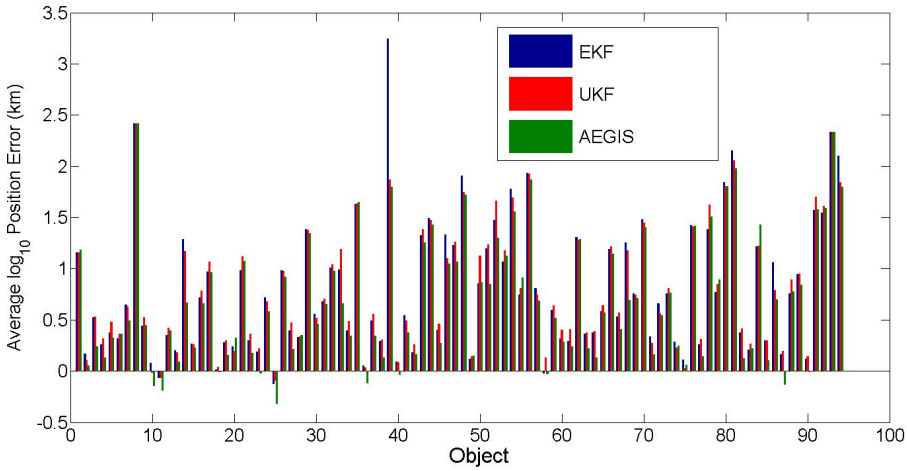

Figure 5.56. Bar graphs showing the $\log_{10}$ average position error for each object given in Eq. 5.7, through implementation of an EKF, UKF and AEGIS filter in conjunction with an SIG tasking strategy.

As shown in Figure 5.57, when using SIG tasking there was a more even distribution of observations as opposed to the FIG simulation, resulting in less objects accumulating large estimate errors than was experienced using FIG tasking. Once again, this followed the same trend as in the low-error simulation, except that

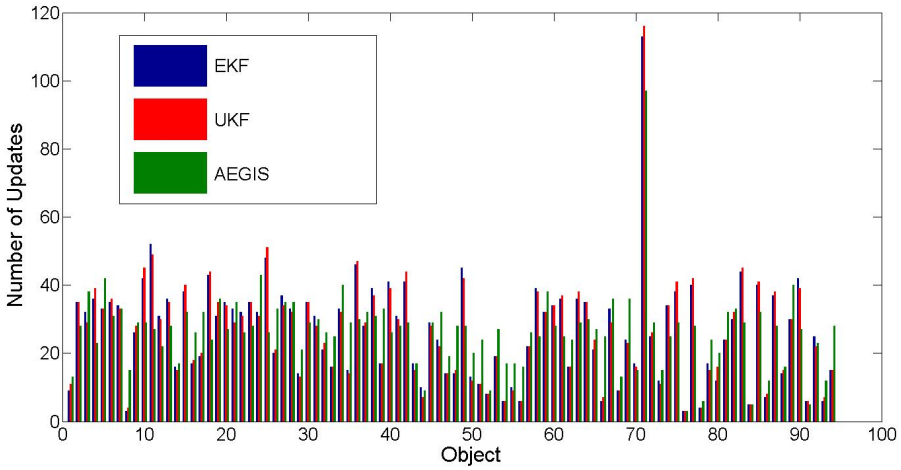

Figure 5.57. Bar graphs showing the updates each object received through implementation of an EKF, UKF, and AEGIS filter in conjunction with an SIG tasking strategy.

there were no objects which the AEGIS filter produced a much larger magnitude of observations than the EKF or UKF. However, the AEGIS filter still produced much different tasking schedules for many objects when compared to the number of updates those object received using the EKF or UKF. In some cases, the AEGIS filter resulted in much more updates than the EKF or UKF (e.g. objects 8, 16, 17, etc.) while for others it produced fewer (e.g. objects 4, 10, 11, 71, etc). As with the low-error simulation, these different tasking schedules typically resulted in lower average position errors than the EKF or UKF, pointing to a greater degree of tasking efficiency when using the AEGIS filter.

While the UKF provided a slight overall performance advantage to the EKF, Figure 5.56 shows that the EKF produced lower average position estimate errors for several objects. Much like in the low-error simulation, the difference was that for objects which accumulated the largest errors, the EKF would have an equal or greater error than the UKF or AEGIS filter (e.g. objects 8, 58, 81, and most notably object 39), which would have resulted in a higher overall average estimated position error in Table 5.13. Furthermore, Figure 5.56 shows that unlike in the low-error test case, the UKF did not consistently produce lower average estimated position errors for the best performing objects than the EKF.

Also, in a similar manner to the low-error simulation simulation, the AEGIS filter would generally provide equal or more updates to objects which were available for few observations (e.g. objects 1, 8, 93, etc), and less observations for objects which were often available (most notably object 71). This again points to the more efficient sensor schedules using the AEGIS filter, in that it can use less observations to create equal or better average position errors for objects frequently available for observation, allowing for more observations to objects sparsely available to any sensors.

To investigate why sensor tasking differed between the EKF, UKF, and AEGIS filter when implementing SIG, Figures 5.58 - 5.63 show when object 39 (the worst performing SIG object using either filter) was available for observation, tasked for observation, as well as the time histories of its position estimate error using SIG, the position error in the all data case, its maximum SIG utility metric (at times it was available for observation), and the average SIG metric for objects tasked for observation. Figures 5.61 - 5.63 show that object 39, while having many available windows for observation, experiences 3 long periods when no observations were possible (roughly 5,000 - 50,000 sec, 51,000 - 99,000 sec, and 100,000 - 135,000 sec). The major differences between the filters in this case was that the UKF and AEGIS filter provided better updates (i.e. decreased position error more) than the EKF, and that the AEGIS filter provided much more observations than the EKF or UKF (17 for the EKF and UKF, 33 for the AEGIS filter). With respect to providing better updates, Figures 5.62 - 5.63 show that the UKF and AEGIS filter can recover easier from a position error which has diverged to a large extent, going from a position error of roughly 1000 km to one which is just below 10 km. While these errors are too high for any practical application of satellite tracking, they illustrate in general terms the benefit a better filter can provide, in that its easier to recover from a diverging estimation error using these filters as opposed to an EKF (which comparatively went from about 1000 km to about 100 km).

When observing Figures 5.61 - 5.63, the SIG metric for object 39 varies little between the EKF and UKF filters, while there is a noticeable difference using the AEGIS filter, as was the case in the low-error test case. Once again, these

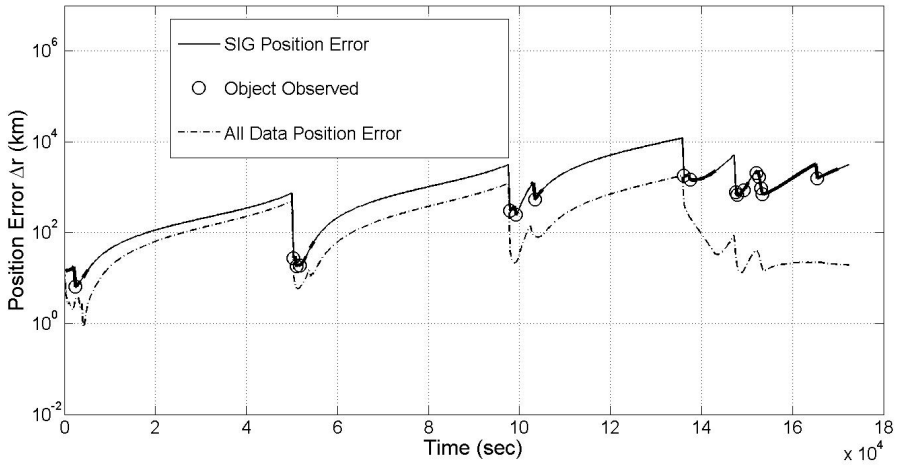

Figure 5.58. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 39 using SIG tasking strategy in conjunction with an EKF. Thick sections of line reflect times when observations were available between object 39 and one or more sensors.

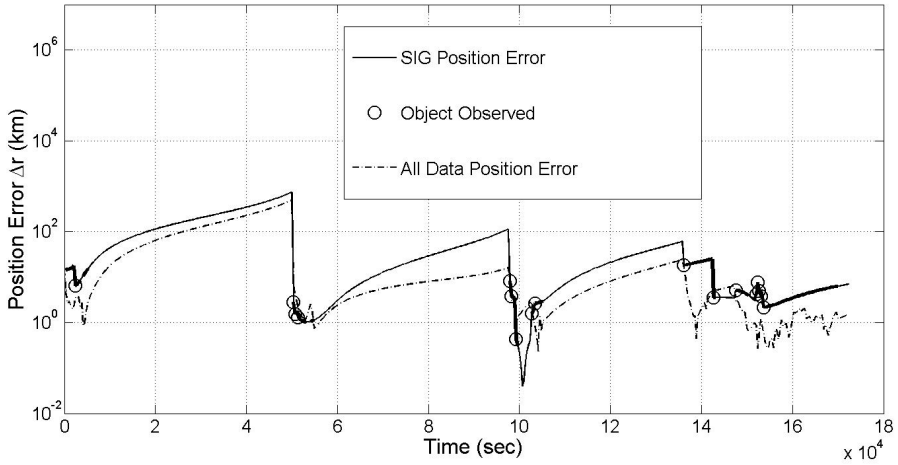

Figure 5.59. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 39 using SIG tasking strategy in conjunction with a UKF. Thick sections of line reflect times when observations were available between object 39 and one or more sensors.

differences are that the mean SIG for tasked objects decreases to a larger extent throughout the simulation for the AEGIS filter, and the SIG metric for object 39 remains closer to the mean for the AEGIS filter than the EKF or UKF, resulting in more observations using the AEGIS filter. As was the case for the low-error simulation, this can be explained by observing the time history of the average estimated error ellipse metric in Figure 5.64. As a reiteration from the low-error

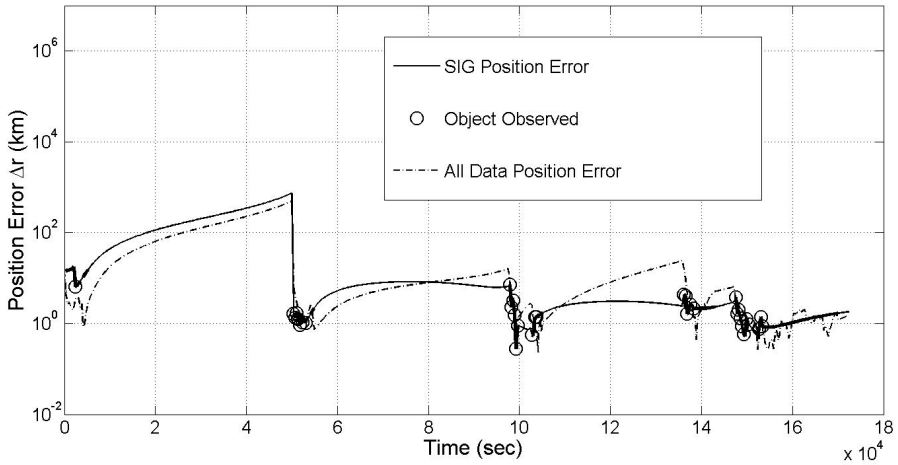

Figure 5.60. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 39 using SIG tasking strategy in conjunction with an AEGIS filter. Thick sections of line reflect times when observations were available between object 39 and one or more sensors.

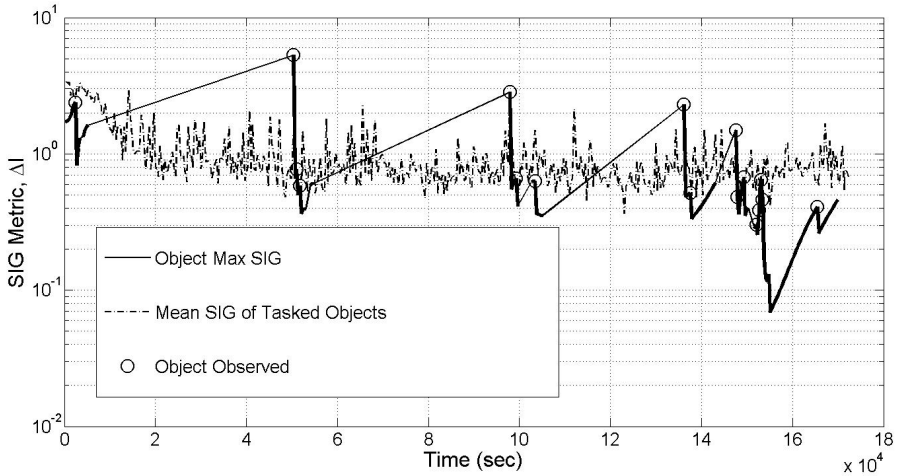

Figure 5.61. Plot of the SIG utility metric ΔI and tasking decisions for object 39 using SIG tasking strategy in conjunction with an EKF. Thick sections of line reflect times when observations were available between that object and one or more sensors.

simulation, the AEGIS filter typically gets much better covariance updates resulting in a lower value of $\hat{A}_{err}$ when updates occur, and therefore less divergence in $\hat{A}_{err}$ in subsequent time steps when updates do not occur. Since the SIG metric relies on the ratio of determinants of covariance estimates between the forecast and update steps, if this change is greater for the AEGIS filter than other filters, it will result in a different evolution in SIG metric as well.

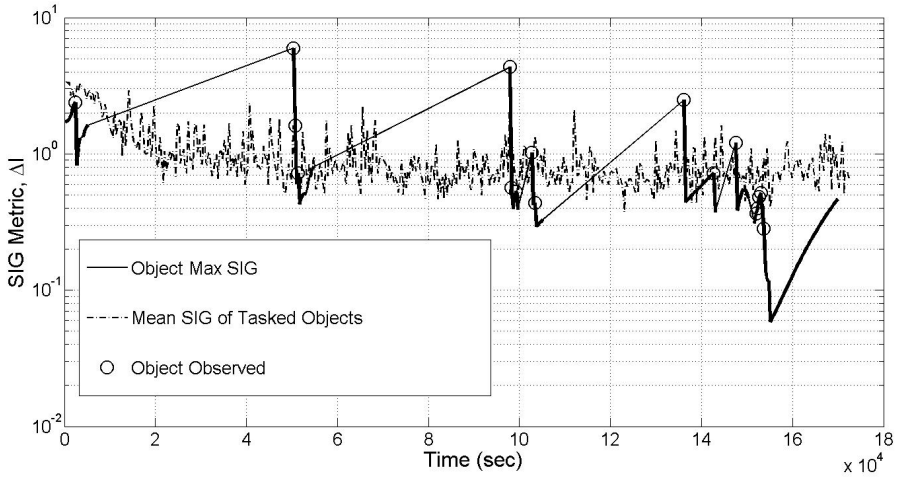

Figure 5.62. Plot of the SIG utility metric ΔI and tasking decisions for object 39 using SIG tasking strategy in conjunction with a UKF. Thick sections of line reflect times when observations were available between that object and one or more sensors.

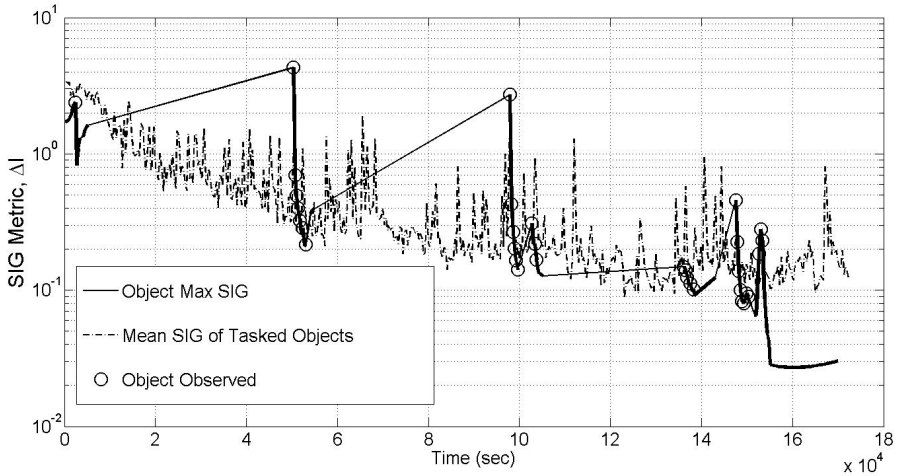

Figure 5.63. Plot of the SIG utility metric ΔI and tasking decisions for object 39 using SIG tasking strategy in conjunction with an AEGIS filter. Thick sections of line reflect times when observations were available between that object and one or more sensors.

Also of interest is the evolution of the value for $\hat{A}_{err}$ for the UKF as compared to the EKF. In much the same way as Figure 5.41, the EKF often produces a lower value of $\hat{A}_{err}$ for much of the simulation than the UKF, a result which goes counter to its higher position errors as seen in Figure 5.58. This once again illustrated why the EKF can be overconfident in its uncertainty estimates, resulting in a higher

percentage of position errors $\Delta r_{i,k} \geq 3\hat{\sigma}^r_{i,k}$ and overall worse performance than the UKF.

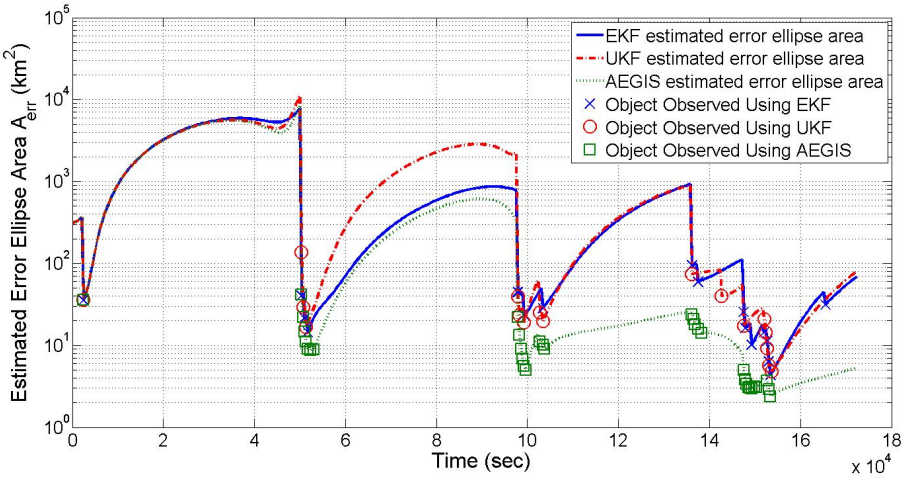

Figure 5.64. EKF, UKF, and AEGIS plots of $\hat{A}_{err} = \pi\sqrt{\hat{\varsigma}^1_{i,k}\hat{\varsigma}^2_{i,k}}$ metric for object 39 in the SIG simulation.

5.4.4 LLE Tasking

Table 5.14 shows the scalar performance metrics outlined in Eqs. 5.5 and 5.8 for the high-error simulation using LLE tasking. In addition, figures showing the distribution of position errors and how these errors compared to the estimated $3\hat{\sigma}^r_{i,k}$ provided by the filter covariance estimates are presented in Figures 5.65 - 5.66.

Table 5.14. Simulation Performance Metrics for LLE High-Error Simulation

Filter	$\langle E_r^* \rangle$ (km)	$\left\langle \hat{A}_{err} \right\rangle$ (km^2)	$\frac{\langle E_r^* \rangle_{Tasking}}{\langle E_r^* \rangle_{All\ Data}}$	$\frac{\langle \hat{A}_{err} \rangle_{Tasking}}{\langle \hat{A}_{err} \rangle_{All\ Data}}$
EKF	21.451	1.131×10^3	1.163	1.188
UKF	17.249	1.042×10^3	1.145	1.231
AEGIS	16.193	9.463×10^2	1.169	1.205

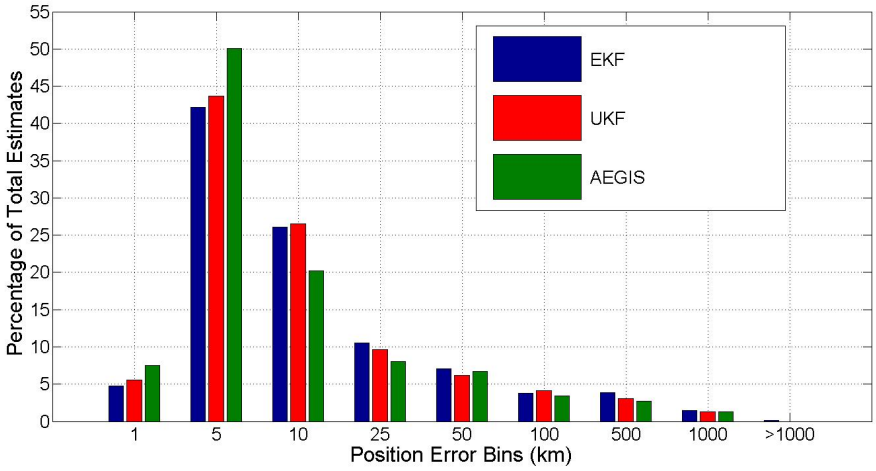

Figure 5.65. EKF, UKF, and AEGIS histograms of position estimate error distributions for LLE simulation. Plots include data from all objects at all simulation time steps.

When observing the performance utilizing LLE tasking, it provides better estimates for all filters than the FIG strategy, but in contrast to the low-error simulation also results in better performance than the SIG method. This benefit is most prevalent in the EKF, which showed significant improvement in overall performance when compared to the SIG simulation, while the AEGIS filter showed only a slight improvement. From observing Figures 5.65 - 5.66, though the LLE tasking strategy results in less objects within position estimate errors of 1 km than the SIG method, it also results in less greater than 50 km, which accounted for

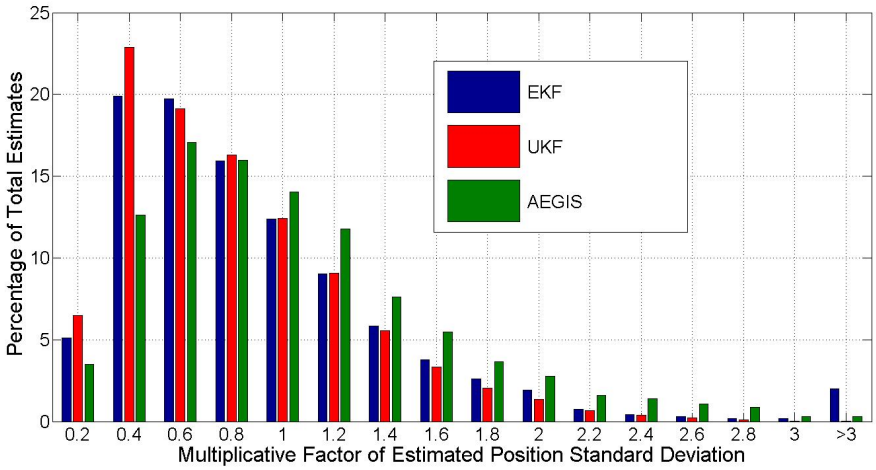

Figure 5.66. EKF, UKF, and AEGIS histograms of $J^{\hat{P}}_{i,k}$ metric in Eq. 5.9 for LLE simulation. Plots include data from all objects at all simulation time steps.

most of the improved performance when compared to the SIG simulation. Additionally, while the EKF had a large improvement in its average position estimate error, it also produced roughly the same percentage of position estimate errors $\Delta r_{i,k} \geq 3\hat{\sigma}^r_{i,k}$ as the SIG simulation. Additionally, while the AEGIS filter did not show a drastic improvement in performance over the SIG method, there was an increase position estimate errors in the range $0.8\hat{\sigma}^r_{i,k} \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}^r_{i,k}$ than in the SIG simulation, indicating that the AEGIS filter provided slightly more accurate uncertainty estimates using the LLE tasking strategy.

When compared to results in the low-error simulation, the AEGIS filter produced less position estimates outside the $\Delta r_{i,k} \geq 3\hat{\sigma}^r_{i,k}$ bounds, while all filters produced more accurate estimates in the range of $0.8\hat{\sigma}^r_{i,k} \leq \Delta r_{i,k} \leq 1.2\hat{\sigma}^r_{i,k}$. This is interesting in that all of the filters produced more accurate uncertainty estimates in the high-error estimation than in the low-error estimation, a fact that aided in the superior performance of the LLE method over the SIG method in the high-error simulation. Furthermore, the three filters had a greater discrepancy between performance metrics in the high-error case, with the difference between the EKF and UKF being higher than that of the UKF and AEGIS filter. Again, as with the FIG and SIG simulations, the high-error simulation shows that the EKF is more susceptible to poor performance due to an increase in simulation error than the

UKF or AEGIS filter.

To investigate why performance discrepancies existed in the implementation of EKF, UKF, and AEGIS filters in conjunction with an LLE tasking strategy, Figures 5.67 - 5.68 show the amount of updates each simulated object received along with their corresponding average position errors in the LLE simulation.

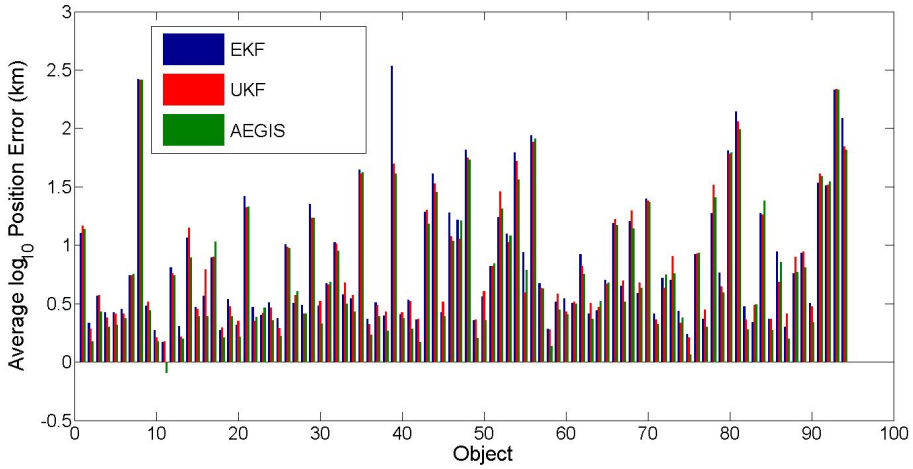

Figure 5.67. Bar graphs showing the $\log_{10}$ average position error for each object given in Eq. 5.7, through implementation of an EKF, UKF and AEGIS filter in conjunction with an LLE tasking strategy.

Once again, as was the case with the low-error simulation, Figure 5.68, shows a more even distribution of observations among all the simulated objects than the FIG method, and is more similar to the SIG method. There does exist a slight difference in the LLE and SIG methods, in that the LLE method resulted in slightly better performance to several of the worst performing objects (e.g. objects 1, 16, 17, 91, 92, etc. and especially object 39), while resulting in slightly worse performance for objects which performed relatively well (e.g objects 10, 11, 18, 25, etc.). Also in a similar manner to the SIG method and LLE low-error simulation, while several objects have a minor difference in the amount of updates received using an EKF or UKF, the AEGIS filter provides much different tasking strategies for many objects. For the objects with the worst position estimate error (i.e. objects 8, 35, 39, 44, etc.) the EKF either had an equal or slightly worse (the only

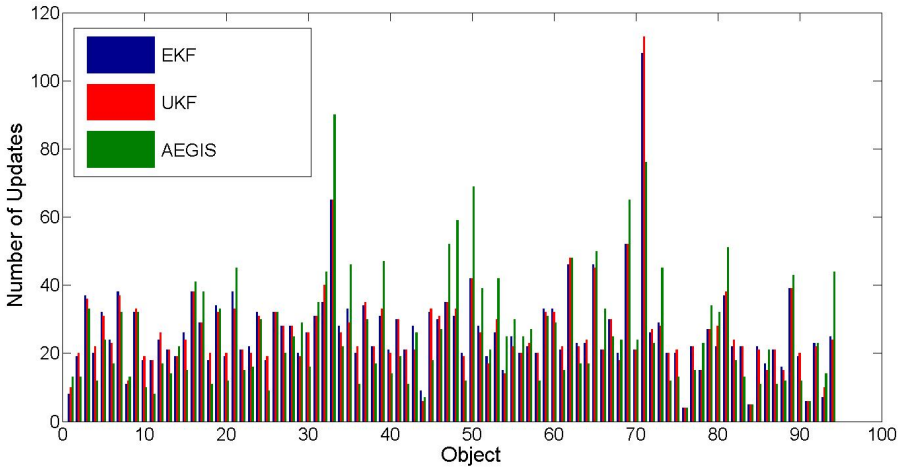

Figure 5.68. Bar graphs showing the updates each object received through implementation of an EKF, UKF, and AEGIS filter in conjunction with an LLE tasking strategy.

case where the position estimate error was drastically worse was for object 39) average position estimate error than the UKF, and would receive equal or slightly less updates. This indicates that while the tasking decisions between the filters are not as obvious as with using FIG-based tasking, slight discrepancies still exist which show the UKF typically provides more observations to the worse performing objects than the EKF, resulting in more efficient tasking decisions. For many of these objects, the AEGIS filter would provide more updates than either the EKF or UKF, often resulting in a better average position estimate error. In much the same way as the SIG method, the LLE tasking using the AEGIS filter would also typically result in less observations for objects with the best average position estimate errors than the EKF or UKF, while generally resulting in comparable to improved performance (e.g. objects 10, 11, 13, 18, etc.).

To examine the LLE metric, and how it effects tasking decisions between the EKF, UKF, and AEGIS filter, Figures 5.69 - 5.74 show when object 39 (the worst performing LLE object using either filter) was available for observation, tasked for observation, as well as the time histories of its position estimate error using LLE, the position error in the all data case, its maximum LLE utility metric (at times it was available for observation), and the average LLE metric for objects tasked for observation.

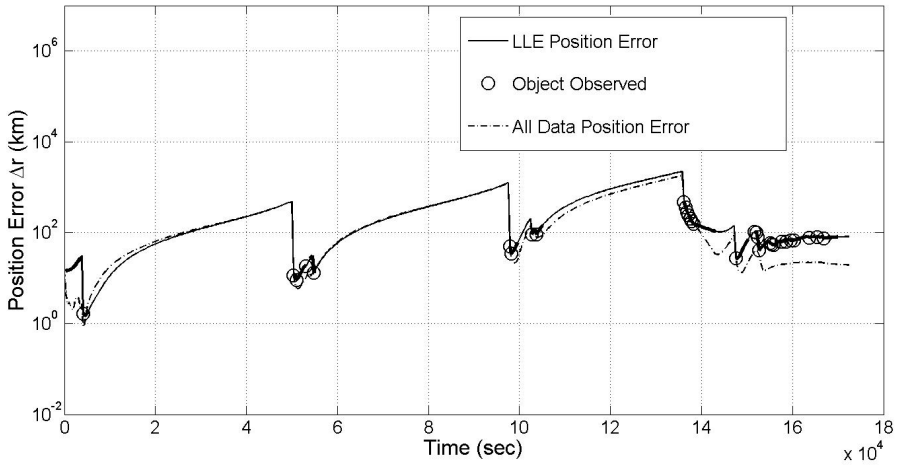

Figure 5.69. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 39 using LLE tasking strategy in conjunction with an EKF. Thick sections of line are times at which observations were available between object 39 and one or more sensors.

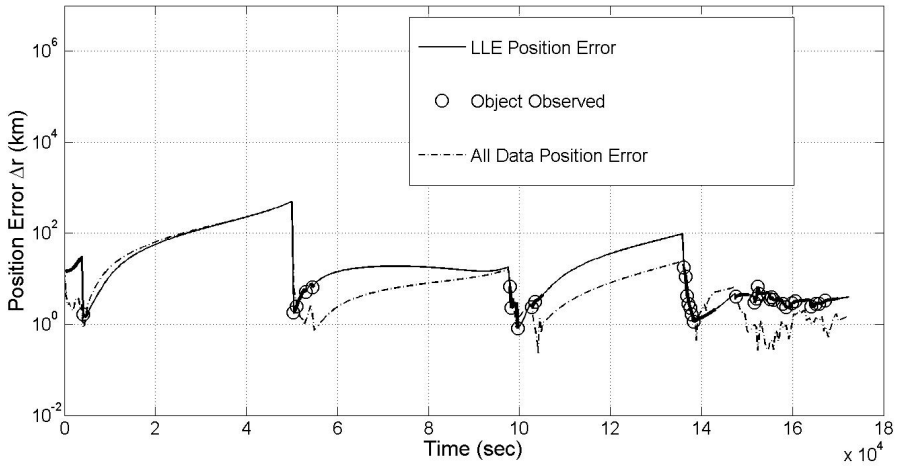

Figure 5.70. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 39 using LLE tasking strategy in conjunction with a UKF. Thick sections of line are times at which observations were available between object 39 and one or more sensors.

Figures 5.69 - 5.70 show that the worst object in the LLE simulation was one that experienced long time spans when observations were not possible, the largest of these periods between times of approximately 5,000 - 50,000 seconds, 55,000 -

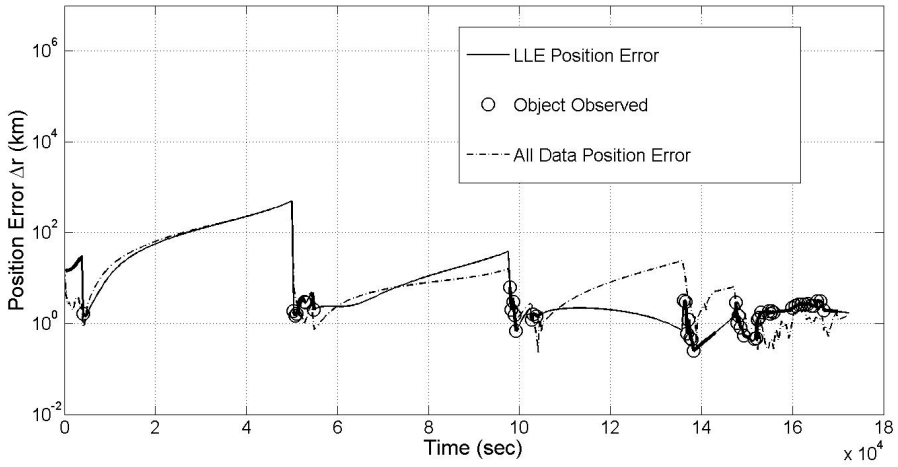

Figure 5.71. Plot of the position error $\Delta r_{i,k}$ and tasking decisions for object 39 using LLE tasking strategy in conjunction with an AEGIS filter. Thick sections of line are times at which observations were available between object 39 and one or more sensors.

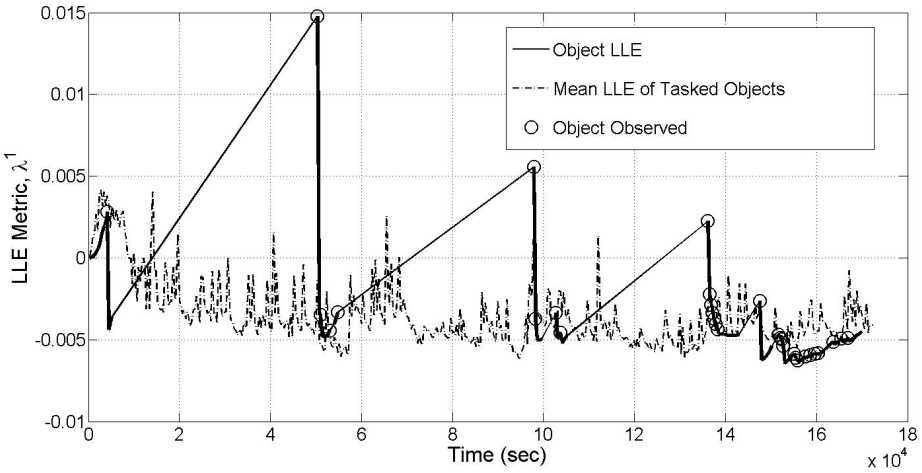

Figure 5.72. Plot of the LLE utility metric $\hat{\lambda}^1$ and tasking decisions for object 39 using LLE tasking strategy in conjunction with an EKF. Thick sections of line are times at which observations were available between that object and one or more sensors.

95,000 seconds, and 105,000 - 135,000 seconds. In each case, the object was immediately observed at the first instance it was available, which was one trait the LLE metric was hypothesized to provide, as was the case in the low-error simulation. During these time spans, the position estimate error would begin to diverge, as was expected. However, while both diverged, they did so at different rates, particularly in the time span of 55,000 - 95,000 seconds. This was largely due to the

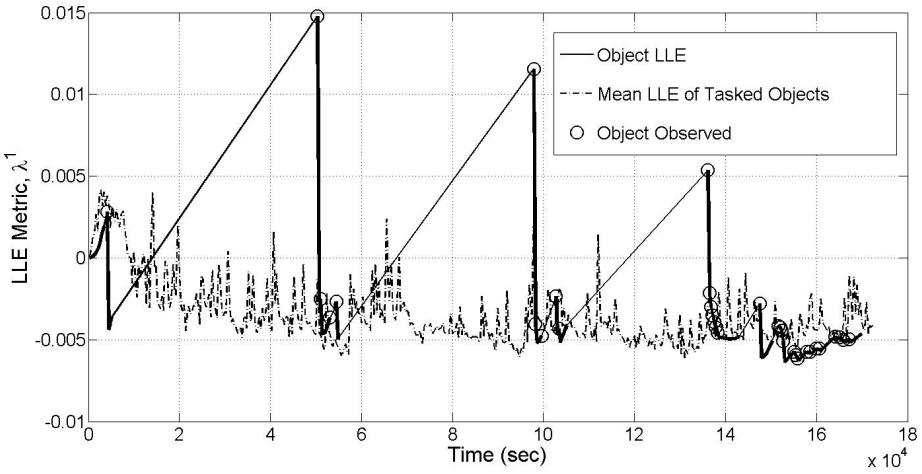

Figure 5.73. Plot of the LLE utility metric $\hat{\lambda}^1$ and tasking decisions for object 39 using LLE tasking strategy in conjunction with a UKF. Thick sections of line are times at which observations were available between that object and one or more sensors.

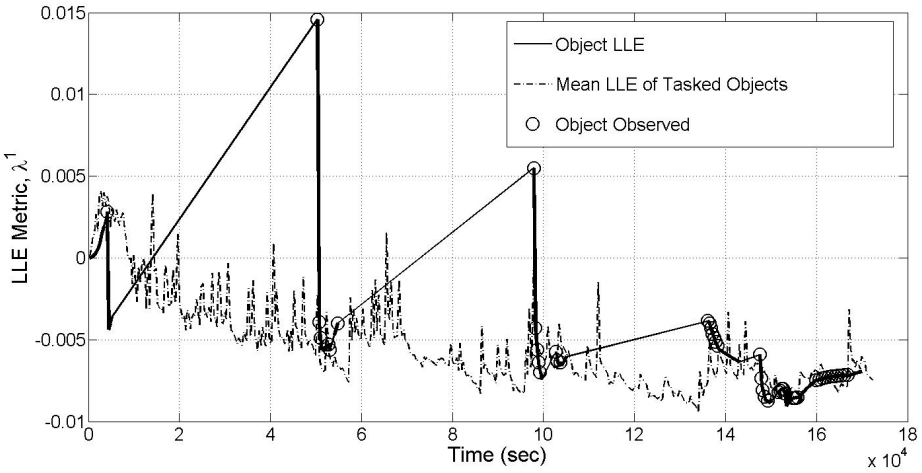

Figure 5.74. Plot of the LLE utility metric $\hat{\lambda}^1$ and tasking decisions for object 39 using LLE tasking strategy in conjunction with an AEGIS filter. Thick sections of line are times at which observations were available between that object and one or more sensors.

better updates the UKF and AEGIS filter received as opposed to the EKF at approximately 50,000 seconds, as illustrated by the larger drop in position estimate errors. After these updates, the error from the UKF and AEGIS filter did not diverge at the same rate as the EKF. Due to this, once observations were again possible around 95,000 seconds, the position error using the UKF and AEGIS filter were considerably less than the EKF. Since object 39 had equal updates at equal

times using both the EKF and UKF over these time spans, the difference in performance can be directly attributed to the filter selection, and not a filter/tasking coupling effect. However, at the time shortly after 95,000 seconds observations were once again available for a short period of time, for which the EKF received 2 and the UKF received 3. This slight difference of one observation follows the trend that for the objects with the worst estimation errors, the UKF resulted in slightly more observations than the EKF, as was observed in Figure 5.68. A similar additional update occurred near the end of the simulation at approximately 165,000 seconds, where the UKF received 4 and the EKF 3. The result, both from the two additional updates gained from the UKF (there were 31 total for the EKF, 33 for the UKF) as well as the UKF's more effective updates was that the UKF consistently had a lower position estimate error than the EKF, and ended on a final value approximately one order of magnitude lower.

Figure 5.71 differs from Figures 5.69 - 5.70 primarily in how many more updates object 39 received using the AEGIS filter as opposed to the EKF or UKF (47 observations compared to 31 for the EKF and 33 for the UKF). Like in the low-error test case, this was much higher than what object 39 received for either the EKF or UKF, and follows the same trait that the AEGIS filter provides a disproportionate amount of updates to the worst performing objects, resulting in more efficient tasking compared to the EKF or UKF.

Figures 5.72 - 5.74 show that the LLE metric has a strong correlation to the actual estimate error as the simulation progresses (i.e. rises and falls with estimate error) as it did in the low-error simulation. In addition, as was the case for the SIG simulation and LLE low-error simulation, the LLE metric for the AEGIS filter behaves slightly different than the EKF or UKF, in that its average value of tasked objects decreases more over time than for the EKF or UKF. As explained in the SIG high-error simulation, this is the result of the advantages in obtaining the covariance estimate using the AEGIS filter as opposed to the EKF or UKF. The reason for this is the same as it was in the low-error LLE simulation, in that the covariance estimates are decreasing more using the AEGIS filter, where for the EKF and UKF they are more stabilized. This results in the worst objects being

tasked for more observations using the AEGIS filter because the average LLE value among all objects decreases to a greater extent. As with the SIG simulation, these observations point to the AEGIS filter providing not only an estimation advantage, but also a tasking advantage using the LLE strategy as opposed to the EKF or UKF. When compared to the SIG high-error simulation, the higher initial errors and sensor noise result in more divergence in estimation errors for poorly performing objects. Since the LLE metric is supposed to tasked objects for observation based upon the divergence of these estimation errors, it is better at avoiding this divergence than the SIG method, resulting in the improved performance with respect to the SIG method in the high-error simulations.

Chapter 6

Conclusions

In these studies, the relationship between nonlinear state/uncertainty estimation and dynamic sensor tasking was investigated as applied to a simplified two-dimensional satellite tracking problem. Estimator and tasking combinations were tested against two of these tracking scenarios, one in which initial estimation errors and sensor noise were low (reflecting ideal tracking conditions) while the other had higher initial errors and sensor noise (highlighting how these combinations fare in non-ideal conditions). The estimators used were an extended Kalman filter (EKF), an unscented Kalman filter (UKF), and an adaptive entropy-based Gaussian information synthesis (AEGIS) filter, while the tasking methods were based on information-theoretic approaches using Fisher information gain (FIG), Shannon information gain (SIG), and a new stability approach using largest Lyapunov exponent estimation (LLE). Simulation results showed that not only was the discrepancy between the three estimators drastic at times, but that the tasking methods based on LLE and SIG in combination with an AEGIS filter provided the most accurate estimates, while FIG based tasking with an EKF provided the worst.

When comparing the three methods of tasking, in both the high-error and low-error simulations the implementation of FIG tasking consistently resulted in the worst performance for all filters with respect to either the SIG or LLE tasking methods. This is primarily due to the FIG-based tasking failing to distribute sensor observations as well among all the satellites, causing a more rapid divergence some of the satellite's estimation errors. This poor distribution is both a result of

FIG being myopic in its implementation, as well as it being a measure of absolute (and therefore unscaled) information gain, leading to objects being observed based on the information increase between an object-sensor pair at a single simulation time step. Consequently, an object-sensor pair that consistently has a low value of FIG with respect to other objects will rarely be observed, while ones that have a high FIG will be observed very frequently. This differed from the implementation of SIG tasking, which while still myopic, was a relative measure of information, meaning that even though an observation of a particular object may reflect a low total gain in information with respect to others, if the gain in information is a substantial improvement to its current information state the object will have a high priority for observation. This resulted in a more even distribution of observations for SIG tasking, which was the best performing tasking method for all filters in the low-error simulation.

However, in the high-error simulation for all three filters, the LLE-based tasking provided more accurate estimates of both the satellite state and uncertainty. The LLE metric is based on assessing the stability of the filter-specific error dynamics over the total simulation time, such that objects that have diverging estimates of uncertainty gain higher priorities for observation. While this performed better than the SIG tasking in the high-error simulation, it did not do so in the low-error simulation. This was because in the low-error simulation, divergence of estimation error was not as much of a concern than in the high-error simulation, meaning that in the low-error simulation a myopic approach had little consequences, and therefore the SIG approach resulted in the best performance. In contrast, not observing an object with tendencies towards diverging estimation errors in the high-error simulation could have much greater consequences, and therefore the LLE method tasked to avoid these divergences, and resulted in better performance for several objects which had the greatest divergence in estimation error. However, while the LLE method was beneficial for those few objects, the myopic SIG method provided better performance for objects with lower estimation errors. This points to the LLE tasking method being advantageous in tracking situations when estimation errors are expected to diverge, while an SIG approach would be better when estimation errors are expected to remain more stable, or when objects are expected

to maintain a low degree of uncertainty.

When comparing the three estimators used, the AEGIS filter consistently had the best performance for any tasking strategy, followed by the UKF and finally the EKF with the worst. One of the clearest examples of this filter discrepancy (and therefore an example of a filter-tasking coupling effect) was with the FIG tasking, and results from how the FIG is calculated based on the filter implemented. For an EKF, it will be based solely on sensor noise and partials of the nonlinear measurement function, and for a UKF of AEGIS filter it will be based on the sensor noise and distribution of sigma points. This resulted in the extremely poor performance seen with the FIG-EKF combination. While performance discrepancies were much less extreme for the other filter/tasking combinations, discrepancies still existed, and were generally more dramatic for the high-error simulations than the low-error simulations. In particular, much of these discrepancies were the result of the difference in covariance propagation and updating between the three filters. In general, the EKF received the worst updates (i.e. obtained the smallest reduction in uncertainty during an update) while the AEGIS received the best, a fact which contributed mightily to the discrepancies in sensor tasking between the three filters. In these discrepancies, the EKF generally resulted in the most inefficient tasking strategies, followed by the UKF and AEGIS filters. These inefficient tasking decisions were highlighted in the SIG and LLE tasking strategies, where the EKF would produce slightly less updates to the worst performing objects than the UKF or AEGIS filter. In these cases, the UKF produced slightly more updates while the AEGIS filter produced at times over twice as much. This was due to the AEGIS filter providing superior updates during observation, meaning it could use less updates to obtain a satisfactory degree of uncertainty for many objects, leaving a surplus of updates to be used on the worst performing objects, and therefore resulting in the most efficient tasking strategies.

While these studies illustrated that a coupling effect is present between estimator type and tasking strategies (should those strategies be covariance-based), this work can be furthered by incorporating different nonlinear filters, and by examining the coupling between estimation and data association. In the former, particle

filters could be implemented. While computationally expensive, these filters can generate non-Gaussian mean and covariance estimates which describe the location and uncertainty of an object using no approximations. The benefit of this could be that a particle filter can provide more accurate estimates of tasking metrics, possibly providing better performance as opposed to filters which are constrained to have Gaussian PDF's. However, a setback to particle filters is that while they can describe a non-Gaussian mean and covariance, they give no indication of the PDF those estimates describe, unlike the AEGIS filter which does both.

In the latter case, including data association would help investigate a very important component in satellite tracking problems which was purposefully omitted from these studies. Since this work was concerned with the coupling between estimation and covariance-based sensor tasking, incorporating the process of data association would have introduced an unnecessary variable to this work. However, since many data association techniques use uncertainty estimates to validate whether observations are being taken on one object as opposed to another (in close proximity), the work done in these studies could translate into examining coupling between estimation, sensor tasking, and data association. The objective of such research would be to show how the process of satellite tracking can be improved by the application of better filters, not only in the estimation component, but in other factions of the problem which use estimates in their implementation. This would show the effects of synergy between different subsets of satellite tracking problems such as space situational awareness, and could be a benefit in the advancement of this very important problem comprising elements of spacecraft mission modeling, conjunction analysis, space object characterization, validation, orbit determination, and sensor network management.

Appendix A

The Determinant of the FIG Matrix for an EKF

Consider a simple EKF filtering problem containing a two-state equation of motion

$$\vec{x} = [x_1, x_2]^T \tag{A.1}$$

with two measurement functions

$$\vec{y} = [h_1(\vec{x}), h_2(\vec{x})]^T + [v_1, v_2]^T \tag{A.2}$$

and measurement noise covariance

$$R = \begin{bmatrix} v_1^2 & 0 \\ 0 & v_2^2 \end{bmatrix} \tag{A.3}$$

From application of Eq. (4.31), the FIG matrix is

$$\hat{\Omega} = \begin{bmatrix} \partial h_1/\partial x_1 & \partial h_2/\partial x_1 \\ \partial h_1/\partial x_2 & \partial h_2/\partial x_2 \end{bmatrix} R^{-1} \begin{bmatrix} \partial h_1/\partial x_1 & \partial h_1/\partial x_2 \\ \partial h_2/\partial x_1 & \partial h_2/\partial x_2 \end{bmatrix} \tag{A.4}$$

which if simplified yields

$$\hat{\Omega} = \begin{bmatrix} \left(\frac{\partial h_1}{\partial x_1}\right)^2 \frac{1}{v_1^2} + \left(\frac{\partial h_2}{\partial x_1}\right)^2 \frac{1}{v_2^2} & \frac{\partial h_1}{\partial x_1}\frac{\partial h_1}{\partial x_2}\frac{1}{v_1^2} + \frac{\partial h_2}{\partial x_1}\frac{\partial h_2}{\partial x_2}\frac{1}{v_2^2} \\ \frac{\partial h_1}{\partial x_1}\frac{\partial h_1}{\partial x_2}\frac{1}{v_1^2} + \frac{\partial h_2}{\partial x_1}\frac{\partial h_2}{\partial x_2}\frac{1}{v_2^2} & \left(\frac{\partial h_1}{\partial x_2}\right)^2 \frac{1}{v_1^2} + \left(\frac{\partial h_2}{\partial x_2}\right)^2 \frac{1}{v_2^2} \end{bmatrix} \tag{A.5}$$

Taking the determinant of Eq. (A.5) results in

$$\left|\hat{\Omega}\right| = \begin{array}{c} \left(\left(\frac{\partial h_1}{\partial x_1}\right)^2\left(\frac{\partial h_1}{\partial x_2}\right)^2\frac{1}{v_1^2} + 2\left(\frac{\partial h_1}{\partial x_1}\right)^2\left(\frac{\partial h_2}{\partial x_2}\right)^2\frac{1}{v_1 v_2} + \left(\frac{\partial h_2}{\partial x_1}\right)^2\left(\frac{\partial h_2}{\partial x_2}\right)^2\frac{1}{v_2^2}\right) - \dots \\ \dots - \left(\left(\frac{\partial h_1}{\partial x_1}\right)^2\left(\frac{\partial h_1}{\partial x_2}\right)^2\frac{1}{v_1^2} + 2\left(\frac{\partial h_1}{\partial x_1}\right)^2\left(\frac{\partial h_2}{\partial x_2}\right)^2\frac{1}{v_1 v_2} + \left(\frac{\partial h_2}{\partial x_1}\right)^2\left(\frac{\partial h_2}{\partial x_2}\right)^2\frac{1}{v_2^2}\right) = 0 \end{array} \tag{A.6}$$

Changing this simple example from an n state equation of motion and m observations yields the same results, showing that for an EKF, the determanant of the FIG for these studies is always zero.

Appendix B

Estimation Algorithms

B.1 The Extended Kalman Filter [1]

Step 1: *Initialization (time step $k = 0$)*

State estimate: $\hat{X}^*_{i,0}$ Covariance estimate: $\hat{P}^*_{i,0}$

Step 2: *Forecast* $t \in [t_k, t_{k+1}]$

a) Propagate state estimate from time step $k \to k+1$

$$\hat{X}^f_{i,k+1} = \mathcal{F}\left(\hat{X}^*_{i,k}\right)$$

b) Propagate state transition matrix from time step $k \to k+1$

Integrate $\dot{\Phi}_{i,t_k|t} = \left(\frac{\partial f}{\partial \vec{x}}\right)_{i,t} \Phi_{i,t_k|t}$, $\Phi_{i,t_k|t_k} = I_{n\times n}$

c) Calculate covariance estimate

$$\hat{P}^f_{i,k+1} = \Phi_{i,k|k+1}\hat{P}^*_{i,k}\left[\Phi_{i,k|k+1}\right]^T + Q_{i,k}$$

Step 3: *Update* (Given set of M' sensors to task object i at time $k+1$, $\mathcal{S}^\tau_{i,k+1}$)

a) Calculate measurement partial and estimated measurement

$$H_{i,k+1} = \begin{bmatrix} \partial h_\rho(\vec{x}, \vec{s}_{s_1^\tau,k+1})/\partial\vec{x} \\ \vdots \\ \partial h_\rho(\vec{x}, \vec{s}_{s_{M'}^\tau,k+1})/\partial\vec{x} \\ \partial h_\psi(\vec{x}, \vec{s}_{s_1^\tau,k+1})/\partial\vec{x} \\ \vdots \\ \partial h_\psi(\vec{x}, \vec{s}_{s_{M'}^\tau,k+1})/\partial\vec{x} \end{bmatrix}_{\vec{x}=\hat{X}_{i,k+1}^f}$$

$$\hat{y}_{i,k+1} = \begin{bmatrix} h_\rho\left(\hat{X}_{i,k+1}^f, \vec{s}_{s_1^\tau,k+1}\right) \\ \vdots \\ h_\rho\left(\hat{X}_{i,k+1}^f, \vec{s}_{s_{M'}^\tau,k+1}\right) \\ h_\psi\left(\hat{X}_{i,k+1}^f, \vec{s}_{s_1^\tau,k+1}\right) \\ \vdots \\ h_\psi\left(\hat{X}_{i,k+1}^f, \vec{s}_{s_{M'}^\tau,k+1}\right) \end{bmatrix}$$

b) Calculate sensor noise covariance, cross covariance, innovation covariance, and Kalman gain matrix

$$R_{i,k+1}^\tau = \begin{bmatrix} \left(v_{s_1^\tau,k+1}^\rho\right)^2 & 0 & \cdots & & \cdots & 0 \\ 0 & \ddots & & & & \vdots \\ \vdots & & \left(v_{s_{M'}^\tau,k+1}^\rho\right)^2 & & & \\ \vdots & & & \left(v_{s_1^\tau,k+1}^\psi\right)^2 & & \vdots \\ \vdots & & & & \ddots & 0 \\ 0 & \cdots & & \cdots & 0 & \left(v_{s_{M'}^\tau,k+1}^\psi\right)^2 \end{bmatrix}$$

$$P_{i,k+1}^{xy} = \hat{P}_{i,k+1}^f \left[H_{i,k+1}\right]^T$$

$$P_{i,k+1}^{yy} = H_{i,k+1}\hat{P}_{i,k}^f H_{i,k+1}^T$$

$$K_{i,k+1} = P_{i,k+1}^{xy}\left\{P_{i,k+1}^{yy} + R_{i,k+1}^\tau\right\}^{-1}$$

c) Update state and covariance estimates

$$\hat{X}_{i,k+1}^* = \hat{X}_{i,k+1}^f + K_{i,k+1}[\vec{y}_{i,k+1} - \hat{y}_{i,k+1}]$$

$$\hat{P}^{*}_{i,k+1} = \hat{P}^{f}_{i,k+1} - K_{i,k+1}\left[P^{yy}_{i,k+1} + R^{\tau}_{i,k+1}\right]K^{T}_{i,k+1}$$

d) Update time step $k = k + 1$ and repeat steps 2-3

B.2 The Unscented Kalman Filter [1]

Step 1: *Initialization (time step $k = 0$)*

State estimate: $\hat{X}^{*}_{i,0}$ Covariance estimate: $\hat{P}^{*}_{i,0}$
Sigma point weights:

$$W^{\gamma}_{p} = \begin{cases} \frac{\Lambda}{(n+\Lambda)} + (1 - \alpha^2 + \beta), & \gamma = 0 \\ \frac{1}{2(n+\Lambda)}, & \gamma = 1, \ldots, 2n \end{cases}$$

$$W^{\gamma}_{x} = \begin{cases} \frac{\Lambda}{(n+\Lambda)}, & \gamma = 0 \\ \frac{1}{2(n+\Lambda)}, & \gamma = 1, \ldots, 2n \end{cases}$$

Step 2: *Forecast $t \in [t_k, t_{k+1}]$*

a) Draw and propagate sigma points from time step $k \to k + 1$

$$\mathcal{X}_{i,k} = \hat{X}^{*}_{i,k}\left[1\right]_{1\times(2n+1)} + \sqrt{n+\Lambda}\left[0_{n\times 1}\ \hat{P}^{CH}_{i,k}\ -\hat{P}^{CH}_{i,k}\right]$$
$$\mathcal{X}^{f}_{i,k+1} = \mathcal{F}\left(\mathcal{X}_{i,k}\right)$$

b) Compute state and covariance estimates

$$\hat{X}^{f}_{i,k+1} = \textstyle\sum_{\gamma=0}^{2n} W^{\gamma}_{x}\mathcal{X}^{f,\gamma}_{i,k+1}$$
$$\hat{P}^{f}_{i,k+1} = \textstyle\sum_{2n}^{\gamma=0} W^{\gamma}_{p}\left[\mathcal{X}^{f,\gamma}_{i,k+1} - \hat{X}^{f}_{i,k+1}\right]\left[\mathcal{X}^{f,\gamma}_{i,k+1} - \hat{X}^{f}_{i,k+1}\right]^{T} + Q_{i,k}$$

Step 3: *Update* (Given set of M' sensors to task object i at time $k + 1$, $\mathcal{S}^{\tau}_{i,k+1}$)

a) Calculate measurement sigma points and estimated measurement

$$\mathcal{Y}^{\gamma}_{i,k+1} = \begin{bmatrix} h_{\rho}\left(\mathcal{X}^{f,\gamma}_{i,k+1}, \vec{s}_{s^{\tau}_{1},k+1}\right) \\ \vdots \\ h_{\rho}\left(\mathcal{X}^{f,\gamma}_{i,k+1}, \vec{s}_{s^{\tau}_{M'},k+1}\right) \\ h_{\psi}\left(\mathcal{X}^{f,\gamma}_{i,k+1}, \vec{s}_{s^{\tau}_{1},k+1}\right) \\ \vdots \\ h_{\psi}\left(\mathcal{X}^{f,\gamma}_{i,k+1}, \vec{s}_{s^{\tau}_{M'},k+1}\right) \end{bmatrix}, \ \gamma = 0, \ldots, 2n$$

$$\hat{y}_{i,k+1} = \sum_{\gamma=0}^{2n} W^{\gamma}_{x} \mathcal{Y}^{\gamma}_{i,k+1}$$

b) Calculate sensor noise covariance, cross covariance, innovation covariance, and Kalman gain

$$R^{\tau}_{i,k+1} = \begin{bmatrix} \left(v^{\rho}_{s^{\tau}_{1},k+1}\right)^{2} & 0 & \cdots & & \cdots & 0 \\ 0 & \ddots & & & & \vdots \\ \vdots & & \left(v^{\rho}_{s^{\tau}_{M'},k+1}\right)^{2} & & & \\ \vdots & & & \left(v^{\psi}_{s^{\tau}_{1},k+1}\right)^{2} & & \vdots \\ \vdots & & & & \ddots & 0 \\ 0 & \cdots & & \cdots & 0 & \left(v^{\psi}_{s^{\tau}_{M'},k+1}\right)^{2} \end{bmatrix}$$

$$P^{xy}_{i,k+1} = \sum_{\gamma=0}^{2n} W^{\gamma}_{p} \left[\mathcal{X}^{f,\gamma}_{i,k+1} - \hat{X}^{f}_{i,k+1}\right] \left[\mathcal{Y}^{\gamma}_{i,k+1} - \hat{y}_{i,k+1}\right]^{T}$$

$$P^{yy}_{i,k+1} = \sum_{\gamma=0}^{2n} W^{\gamma}_{p} \left[\mathcal{Y}^{\gamma}_{i,k+1} - \hat{y}_{i,k+1}\right] \left[\mathcal{Y}^{\gamma}_{i,k+1} - \hat{y}_{i,k+1}\right]^{T}$$

$$K_{i,k+1} = P^{xy}_{i,k+1} \left\{P^{yy}_{i,k+1} + R^{\tau}_{i,k+1}\right\}^{-1}$$

c) Update state and covariance estimates

$$\hat{X}^{*}_{i,k+1} = \hat{X}^{f}_{i,k+1} + K_{i,k+1}[\vec{y}_{i,k+1} - \hat{y}_{i,k+1}]$$

$$\hat{P}^{*}_{i,k+1} = \hat{P}^{f}_{i,k+1} - K_{i,k+1}\left[P^{yy}_{i,k+1} + R^{\tau}_{i,k+1}\right] K^{T}_{i,k+1}$$

d) Update time step $k = k + 1$ and repeat steps 2-3

B.3 The Adaptive Entropy-Based Gaussian Information Synthesis Filter [2]

Step 1: *Initialization (time step $k = 0$)*

GMM PDF: $p^g\left(\vec{x}_{i,0}; \hat{X}^*_{i,0}, \hat{P}^*_{i,k}\right) = \sum_{l=1}^{L} \nu^l_{i,0} p^g\left(\vec{x}_{i,0}; \hat{X}^{*,l}_{i,0}, \hat{P}^{*,l}_{i,0}\right)$
Sigma point weights
$W^\gamma_x = W^\gamma_p = \frac{1}{2n},\ \gamma = 1, \ldots, 2n$

Step 2: *Forecast $t \in [t_k, t_{k+1}]$*

a) Draw and propagate sigma points from time step $k \to k+1\ \forall l$
$\mathcal{X}^l_{i,k} = \hat{X}^{*,l}_{i,k}\left[1\right]_{1\times(2n)} + \sqrt{n}\left[\hat{P}^{CH,l}_{i,k}\ \ -\hat{P}^{CH,l}_{i,k}\right]$
$\mathcal{X}^f_{i,k+1} = \mathcal{F}\left(\mathcal{X}^l_{i,k}\right)$

b) If nonlinearity detected at time t, split using data in Table 3.1

c) Compute state and covariance estimates $\forall l$

$\hat{X}^{f,l}_{i,k+1} = \sum_{\gamma=1}^{2n} W^\gamma_x \mathcal{X}^{f,l,\gamma}_{i,k+1}$
$\hat{P}^{f,l}_{i,k+1} = \sum_{2n}^{\gamma=1} W^\gamma_p \left[\mathcal{X}^{f,l,\gamma}_{i,k+1} - \hat{X}^{f,l}_{i,k+1}\right]\left[\mathcal{X}^{f,l,\gamma}_{i,k+1} - \hat{X}^{f,l}_{i,k+1}\right]^T + Q_{i,k}$

Step 3: *Update*

a) Calculate measurement sigma points and estimated measurement $\forall l$

$$\mathcal{Y}_{i,k+1}^{l,\gamma} = \begin{bmatrix} h_\rho\left(\mathcal{X}_{i,k+1}^{f,l,\gamma}, \vec{s}_{s_1^\tau,k+1}\right) \\ \vdots \\ h_\rho\left(\mathcal{X}_{i,k+1}^{f,l,\gamma}, \vec{s}_{s_{M'}^\tau,k+1}\right) \\ h_\psi\left(\mathcal{X}_{i,k+1}^{f,l,\gamma}, \vec{s}_{s_1^\tau,k+1}\right) \\ \vdots \\ h_\psi\left(\mathcal{X}_{i,k+1}^{f,l,\gamma}, \vec{s}_{s_{M'}^\tau,k+1}\right) \end{bmatrix}, \ \gamma = 0, \ldots, 2n$$

$$\hat{y}_{i,k+1}^{l} = \sum_{\gamma=0}^{2n-1} W_x^\gamma \mathcal{Y}_{i,k+1}^{l,\gamma}$$

b) Calculate sensor noise covariance, cross covariance, innovation covariance, and Kalman gain $\forall l$

$$R_{i,k+1}^{\tau} = \begin{bmatrix} \left(v_{s_1^\tau,k+1}^{\rho}\right)^2 & 0 & \cdots & & \cdots & 0 \\ 0 & \ddots & & & & \vdots \\ \vdots & & \left(v_{s_{M'}^\tau,k+1}^{\rho}\right)^2 & & & \\ \vdots & & & \left(v_{s_1^\tau,k+1}^{\psi}\right)^2 & & \vdots \\ \vdots & & & & \ddots & 0 \\ 0 & \cdots & & \cdots & 0 & \left(v_{s_{M'}^\tau,k+1}^{\psi}\right)^2 \end{bmatrix}$$

$$P_{i,k+1}^{xy,l} = \sum_{\gamma=1}^{2n} W_p^\gamma \left[\mathcal{X}_{i,k+1}^{f,l,\gamma} - \hat{X}_{i,k+1}^{f,l}\right]\left[\mathcal{Y}_{i,k+1}^{l,\gamma} - \hat{y}_{i,k+1}^{l}\right]^T$$

$$P_{i,k+1}^{yy,l} = \sum_{\gamma=1}^{2n} W_p^\gamma \left[\mathcal{Y}_{i,k+1}^{l,\gamma} - \hat{y}_{i,k+1}^{l}\right]\left[\mathcal{Y}_{i,k+1}^{l,\gamma} - \hat{y}_{i,k+1}^{l}\right]^T$$

$$K_{i,k+1}^{l} = P_{i,k+1}^{xy,l}\left\{P_{i,k+1}^{yy,l} + R_{i,k+1}^{\tau}\right\}^{-1}$$

c) Update state and covariance estimates and GMM weights $\forall l$

$$\hat{X}_{i,k+1}^{*,l} = \hat{X}_{i,k+1}^{f,l} + K_{i,k+1}^{l}[\vec{y}_{i,k+1} - \hat{y}_{i,k+1}^{l}]$$

$$\hat{P}_{i,k+1}^{*,l} = \hat{P}_{i,k+1}^{f,l} - K_{i,k+1}^{l}\left[P_{i,k+1}^{yy,l} + R_{i,k+1}^{\tau}\right]\left(K_{i,k+1}^{l}\right)^T$$

$$\nu_{i,k+1}^{l} = \nu_{i,k}^{l} p^g\left(\vec{y}_{i,k+1}; \hat{y}_{i,k+1}^{l}, P_{i,k+1}^{yy,l}\right) / \sum_{l'=1}^{L} \nu_{i,k}^{l'} p^g\left(\vec{y}_{i,k+1}; \hat{y}_{i,k+1}^{l'}, P_{i,k+1}^{yy,l'}\right)$$

d) Update time step $k = k + 1$ and repeat steps 2-3

Bibliography

[1] Wan, E. and R. van-der Merwe (2001) *Kalman Filtering and Neural Networks*, chap. 7, Wiley.

[2] DeMars, K. (2010) *Nonlinear Orbit Uncertainty Prediction and Rectification for Space Situational Awareness*, Ph.D. thesis, The University of Texas at Austin, Department of Aerospace Engineering.

[3] Miller, J. G. (2007) "A New Sensor Allocation Algorithm For The Space Surveillance Network," *Military Operations Research*, **12**, pp. 57–70.

[4] Vallado, D. A. and J. D. Griesbach (2011) "Simulating Space Surveillance Networks," in *AAS 11-580, Advances in the Astronautical Sciences*, vol. 142, pp. 2769 – 2787.

[5] (2002) *Space Surveillance Network Optimization (SSNO)*, *Tech. rep.*, B0001-3.1.4-SSNO-RPT-01,Lockheed Martin Mission Systems.

[6] Zatezalo, A., A. El-Fallah, R. Mahler, R. K. Mehra, and J. Brown (2009) "Dispersed and Disparate Sensor Management for Tracking Low Earth Orbit Satellites," in *Signal Processing, Sensor Fusion, and Target Recognition, Proc. of SPIE*, vol. 7336.

[7] Chen, G., H. Chen, K. Pham, E. Blasch, and J. B. Cruz (2009) "Awareness-Based Game-Theoretic Space Resource Management," in *Sensors and Systems for Space Applications III, Proc. of SPIE*, vol. 7330.

[8] Shen, D., G. Chen, K. Pham, and Blasch (2011) "Trust-Based Sensor Allocation Algorithm in Cooperative Space Search Problems," in *Sensors and Systems for Space Applications IV, Proc. of SPIE*, vol. 8044.

[9] Schmaedeke, W. (1993) "Information Based Sensor Management," in *Signal Processing, Sensor Fusion, and Target Recognition, Proc. of SPIE*, vol. 1955.

[10] Kantz, T. and T. Schreiber (2004) *Nonlinear Time Series Analysis*, 2 ed., Cambridge University Press.

[11] Kalandros, M. K., L. Trailovic, L. Y. Pao, and Y. Bar-Shalom (2004) "Tutorial on Multisensor Management and Fusion Algorithms for Target Tracking," in *Proc. of the Amer. Cont. Conf.*, pp. 4734 – 4748.

[12] Hintz, K. and E. McVey (1991) "Multi-Process Constrained Estimation," in *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 21, pp. 237 – 244.

[13] Aughenbaugh, J. and B. La Cour (2011) "Sensor Management for Particle Filter Tracking," in *IEEE Transactions on Aerospace and Electrical Systems*, vol. 47, pp. 503 – 523.

[14] Tian, K. and G. Zhu (2006) "Sensor Management Based on Fisher Information Gain," *Journal of Systems Engineering & Electronics*, **17**(3), pp. 531–534.

[15] Cover, T. M. and J. A. Thomas (2006) *Elements of Information Theory*, 2nd ed., Wiley and Sons Inc.

[16] Kailath, T., A. Sayed, and B. Hassibi (2000) *Linear Estimation*, Prentice Hall, pp. 325–329.

[17] Ristic, B., S. Arulampalam, and N. Gordon (2004) *Beyond the Kalman Filter*, chap. 1-4, Artech House.

[18] Hero, A., D. Castanion, D. Cochran, and K. Kastella (2008) *Foundations and Applications of Sensor Management*, Springer.

[19] Chakravorty, S. and R. Erwin (2011) "Information Space Receding Horizon Control," in *IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning*, pp. 302 – 309.

[20] van Norden, W., J. de Jong, F. Bolderheij, and L. Rothkrantz (2005) "Intelligent Task Scheduling in Sensor Networks," in *7th International Conference on Information Fusion*, pp. 1351–1358.

[21] Erwin, R. S., P. Albuquerque, S. K. Jayaweera, and I. Hussein (2010) "Dynamic Sensor Tasking for Space Situational Awareness," in *Proc. of the Amer. Cont. Conf.*, pp. 1153 – 1158.

[22] Kalman, R. (1960) "A New Approch to Linear Filtering and Prediction Problems," *Transactions of the ASME - Journal of Basic Engineering*, **82**, pp. 35 – 45.

[23] Smith, G., S. Schmidt, and L. McGee (1962) *Application of Statistical Filter Theory to the Optimal Estimation of Position and Velocity on Board a Circumlunar Vehicle*, *Tech. rep.*, Technical Report R-135, NASA.

[24] Kim, J. and S. Sukkarieh (2004) "Autonomous Airborne Navigation in Unknown Terrain Environments," in *IEEE Transactions on Aerospace and Electrical Systems*, vol. 40, pp. 1031 – 1045.

[25] Gelb, A., J. Kasper, R. Nash, C. Price, and A. Sutherland (1992) *Applied Optimal Estimation*, chap. 4,6, The M.I.T. Press.

[26] Julier, S., J. Uhlmann, and H. Durrant-Whyte (2000) "A New Method for the Nonlinear Transformation of Means and Covariances in Filters and Estimators," *IEEE Transactions on Automatic Control*, **45**(3), pp. 477 – 482.

[27] Julier, S. and J. Uhlmann (2004) "Unscented Kalman Filtering and Nonlinear Estimation," in *Proc. of the IEEE*, vol. 92, pp. 401 – 422.

[28] Teixeira, B. O. S., M. A. Santillo, R. S. Erwin, and D. S. Bernstein (2008) "Spacecraft Tracking Using Sampled-Data Kalman Filters," *IEEE Control Systems Magazine*, **28**(4), pp. 78–94.

[29] Lee, D. and K. T. Alfriend (2007) "Sigma Point Filtering for Sequential Orbit Estimation and Prediction," *AIAA Journal of Spacecraft & Rockets*, **44**(2), pp. 388–398.

[30] Langelaan, J. W. (2007) "State Estimation for Autononous Flight in Cluttered Environments," *Journal of Guidance, Control and Dynamics*, **30**(5), pp. 1414 – 1426.

[31] Sorenson, H. and D. Alspach (1971) "Recursive Bayesian Estimation Using Gaussian Sums," *Automatica*, **7**, pp. 465 – 479.

[32] Alspach, D. and H. Sorenson "Nonlinear Bayesian Estimation Using Gaussian Sum Approximations," in *IEEE Transactions in Automatic Control*, vol. AC-17.

[33] Giza, D., P. Singla, and M. Jah (2009) "An Approach for Nonlinear Uncertainty Propagation: Application to Orbital Mechanics," in *AIAA Guidance, Navigation, and Control Conference*.

[34] Terejau, G., P. Singla, T. Singh, and P. Scott (2008) "Uncertainty Propagation for Nonlinear Dynamic Systems Using Gaussian Mixture Models," *Journal of Guidance, Control, and Dynamics*, **31**(6), pp. 1623 – 1633.

[35] DeMars, K., R. Bishop, and M. Jah (2011) "A Splitting Gaussian Mixture Method for the Propagation of Uncertainty in Orbital Mechanics," in *AAS 11-201, Advances in the Astronautical Sciences*.

[36] ——— (2011) "Space Object Tracking in the Presence of Attitude-Dependant Solar Radiation Pressure Effects," in *AAS 11-582, Advances in the Astronautical Sciences*, vol. 142, pp. 2801 – 2820.

[37] Wolf, A., J. Swift, H. Swinney, and J. Vastant (1985) "Determining Lyapunov Exponents from Time Series," *Physica 16D*, pp. 285 – 317.

[38] Rauf, F. and H. A. Ahmed (1991) "Calculation of Lyapunov Exponents Through Nonlinear Adaptive Filters," in *IEEE International Symposium on Circuits and Systems*, vol. 1, pp. 568 – 571.

[39] Vallado, D. (2007) *Fundamentals of Astrodynamics and Applications*, 3 ed., Springer.

[40] Curtis, H. (2005) *Orbital Mechanics for Engineering Students*, Elsevier.

[41] Rao, M. and R. Swift (2006) *Probability Theory with Applications*, chap. 1, 2nd ed., Springer-Verlag.

[42] Papoulis, A. and S. Pillai (2002) *Probability, Random Variables and Stochastic Processes*, chap. 4, 4th ed., McGraw-Hill.

[43] Handeby, G. (2008) *Performance and Implementation Aspects of Nonlinear Filtering*, Ph.D. thesis, Linkoping University, Department of Electrical Engineering.

[44] Weiner, N. (1949) *The Extrapolation, Interpolation and Smoothing of Stationary Time Series*, John Wiley and Sons, pp. 149 – 160.

[45] Bode, H. and C. Shannon (1950) "A Simplified Derivation of Linear Least-Squares Smoothing and Prediction Theory," in *Proceedings IRE*, vol. 38, pp. 417–425.

[46] Doob, J. (1955) *Stochastic Processes*, Wiley.

[47] Loeve, M. (1955) *Probability Theory*, Van Nostrand Company.

[48] Simon, D. (2006) *Optimal State Estimation*, Wiley.

[49] Vergez, P. L. (1997) "Lyapunov Stability Analysis of an Orbit Determination Problem," *The Journal of the Astronautical Sciences*, **45**(2), pp. 233 – 245.

[50] Hanebeck, U. D., K. Briechle, and A. Rauh (2003) "Progressive Bayes: A New Framework for Nonlinear State Estimation," in *Multisensor, Multisource Information Fusion: Architecture, Algorithms and Applications (Proceedings of SPIE)*, vol. 5099, pp. 256–267.

[51] Hanebeck, U. D. (2003) "Progressive Bayesian Estimation for Nonlinear Discrete-Time Systems: The Measurement Step," in *Proceedings of IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems*, pp. 173–178.

[52] Li, Y. and L. Li (2009) "A Split and Merge EM Algorithm for Color Image Segmentation," in *International Conference Intelligent Computing and Intelligent Systems*, pp. 395–399.

[53] ——— (2009) "A Novel Split and Merge EM Algorithm for Gaussian Mixture Model," in *Fifth International Conference on Natural Computation*, pp. 479–483.

[54] Huber, M., T. Bailey, H. Durrant-Whyte, and U. D. Hanebeck (2008) "On Entropy Approximation for Gaussian Mixture Random Vectors," in *Proceedings of IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pp. 181–188.

[55] Maybeck, P. and B. Smith (2005) "Multiple Model Tracker Based on Gaussian Mixture Reduction for Maneuvering Targets in Clutter," in *7th International Conference on Information Fusion*, pp. 40–47.

[56] Yao, Z. and D. Liu (2010) "Gaussian Mixture Reduction Based on KI Divergence," in *Proceedings of the Second International Conference on Future Computer and Communication*, vol. 1, pp. 630–634.

[57] Williams, J. L. and P. S. Maybeck (2003) "Cost-Function-Based Gaussian Mixture Reduction for Target Tracking," in *Proceedings of the Sixth International Conference on Information Fusion*, pp. 1047–1054.

[58] Junkins, J. L., M. R. Akella, and K. T. Alfriend (1996) "Non-Gaussian Error Propagation in Orbital Mechanics," *Journal of the Astronautical Sciences*, **44**(4), pp. 541 – 563.

[59] Park, R. S. and D. J. Scheeres (2007) "Nonlinear Semi-Analytic Methods for Trajectory Estimation," *Journal of Guidance, Control, and Dynamics*, **30**(6), pp. 1688 – 1676.

[60] DeMars, K., M. Jah, and R. Cheng, Y. Bishop (2012) "Methods for Splitting Gaussian Distributions and Applications within the AEGIS Filter,"

in *AAS 12-261, Advances in the Astronautical Sciences*, vol. 143, pp. 2379 – 2398.

[61] Nocedal, J. and S. Wright (2006) *Numerical Optimization*, 2 ed., Springer, pp. 355 – 420.

[62] Shannon, C. (1948) "A Mathematical Theory of Communication," *The Bell System Technical Journal*, **27**, pp. 379–423, 623–656.

[63] Skagerstam, B. (1975) "On the Notions of Entropy and Information," *The Journal of Statistical Physics*, **12**(6), pp. 449–462.

[64] Bourgault, F., A. Makarenko, S. Williams, B. Grocholsky, and H. Durrant-Whyte (2002) "Information Based Adaptive Robotic Exploration," in *IEEE International Conference on Intelligent Robots and Systems*, pp. 540 – 545.

[65] Makarenko, A., S. Williams, F. Bourgault, and H. Durrant-Whyte (2002) "An Experiment in Integrated Exploration," in *IEEE International Conference on Intelligent Robots and Systems*, pp. 534 – 539.

[66] J.R., H. and P. Olsen (2007) "Approximating the Kullback Leibler Divergence Between Gaussian Mixture Models," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 4, pp. IV–317 – IV–320.

[67] Khalil, H. K. (2002) *Nonlinear Systems*, 3 ed., Prentice Hall, p. 111.

[68] Zhang, J. and K. Man (1998) "Time Series Prediction Using Lyapunov Exponents in Embedding Phase Space," in *Proc. of ICSP*, pp. 221 – 224.

[69] Tanaks, T., K. Aihara, and M. Taki (1998) "Analysis of Positive Lyapunov Exponents from Random Time Series," *Phisica D*, **111**, pp. 42 – 50.

[70] Fujisaka, H. (1982) "Multiperiodic Flows, Chaos and Lyapunov Exponents," *Progress of Theoretical Physics*, **68**(4), pp. 1105 – 1121.

Vita

Patrick S. Williams

Patrick Shelton Williams was born in Durham, North Carolina on October 17, 1982. He began his studies in aerospace engineering in the fall of 2001 at the Virginia Polytechnic and State University (Virginia Tech) in Blacksburg, VA. Patrick graduated with a Bachelor of Science degree in May of 2005, followed by a year of employment as a design engineer for Spirit Aerosystems in Wichita, KS. He then took a position as a test engineer for Analytical Graphics Inc (AGI) in the fall of 2006. In the fall of 2007, he enrolled as a graduate student at The Pennsylvania State University in University Park PA, and was granted a research assistantship through funding from AGI. His masters work was based on trajectory optimization, using AGI's software suite Satellite Toolkit (STK) to develop and test optimization algorithms, and was awarded a Master of Science in aerospace engineering in May 2009. Patrick spent the next two years as a teaching assistant for an astrodynamics and spacecraft design class while searching for a Ph.D. topic. In the summers of 2010 and 2011 Patrick worked as a Space Scholar for the Air Force Research Laboratory (AFRL) in Albuquerque NM, specializing in what would eventually be his thesis researching the coupling between nonlinear estimation and dynamic sensor tasking as applied to satellite tracking problems. He spent the spring and fall of 2011, and the spring and summer of 2012 as a research assistant funded by AFRL, while he was an instructor for a spacecraft design class in the fall of 2012.

Permanent address:

1201 W Beaver Ave
State College, PA 16801.