CELESTIAL AIDED INERTIAL NAVIGATION BY TRACKING HIGH ALTITUDE VEHICLES

THESIS

Mark S. Kim, Capt, USAF

AFIT-ENG-MS-17-M-040

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-MS-17-M-040

CELESTIAL AIDED INERTIAL NAVIGATION BY TRACKING HIGH ALTITUDE VEHICLES

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Engineering

Mark S. Kim, B.S.E.E.

Capt, USAF

23 March 2017

AFIT-ENG-MS-17-M-040

CELESTIAL AIDED INERTIAL NAVIGATION BY TRACKING HIGH ALTITUDE VEHICLES

THESIS

Mark S. Kim, B.S.E.E.
Capt, USAF

Committee Membership:

Dr. Stephen C. Cain
Chair

Maj Scott J. Pierce
Member

Dr. John F. Raquet
Member

Abstract

Celestial and inertial navigation systems have been used symbiotically within the area of alternative navigation solutions to Global Positioning System. Celestial systems normally provide attitude updates only by tracking known stars from a catalog, measuring their angular position with respect to the horizon and determining a deviation from the estimated vehicle attitude. However, by imaging reference objects with known positions and velocities against a background field of stars, the celestial system can triangulate its own position and velocity. With the ubiquitous use of aircraft, cooperative vehicles can be a ready source of information from which to make a reference. With a cooperative aircraft, it is assumed that the observer can get continuous updates about the reference vehicle via secure communication links.

This thesis attempts to determine the navigation accuracy of a remotely piloted aircraft using celestial and inertial sensors and a barometric altimeter to track a second aircraft as the reference object. Simulations were performed of the navigating vehicle with fixed star trackers pointed directly up taking observations of the stars and the other aircraft to aid the inertial measurements, which is known to drift. Three different navigating scenarios were studied: 1) using a star sensor to get attitude updates only with just stellar observations, 2) using a star sensor to get position and velocity updates with just reference aircraft observations, and 3) using the the star sensors to get a complete navigation solution with both stellar and reference aircraft observations. The position, velocity, and attitude accuracy are compared between the three scenarios. Additionally, the frequency of celestial navigation measurements is varied between scenarios to determine its effect on navigation accuracy.

The study makes use of an adaptive MATLAB script created by the Air Force

Research Laboratory that simulates a navigating vehicle's flight for which inertial data is generated to provide a dead-reckoning estimate of the vehicle pose. The barometric altimeter provides frequent vertical dimension updates, with additional measurements given by the star trackers. The sensor measurements that are fed into the simulation must be generated externally, which is accomplished as part of this research. The script includes an extended Kalman filter that propagates and updates the navigation state estimates based on the sensor measurements.

The results show that stellar observations alone show a slight improvement in the position, velocity, and attitude estimates, but because the position and velocity are not directly updated, the estimates are still subject to large drift. In the second scenario, observations of the reference vehicle reduce the position error by over 99.9% from stellar observations only with a mean accuracy of 5.93 m. In the third scenario, the combination of both processes provides minor improvement in average position error, down to 5.67 m, but also improves the attitude error uncertainty by 87%. Additionally, the results show that the frequency of measurement updates has little impact on the navigation accuracy due to the use of a navigation grade inertial unit. No significant changes in the navigation estimates were seen from varying the observation frequency from 1 s to 50 s. However at frequencies of 60 s and longer, the measurements are so sparse that other factors interfere with the ability to track the reference aircraft.

Contents

List of Figures

List of Tables

Acronyms

b-frame body frame.

c-frame camera frame.

e-frame Earth-centered Earth-fixed reference frame.

g-frame Geodetic reference frame.

i-frame Earth-centered inertial reference frame.

n-frame navigation frame.

p-frame pixel frame.

3DRMS 3-D root mean square.

CAINS Celestial Aided Inertial Navigation Subsystem.

CDF cumulative distribution function.

CNS Celestial Navigation System.

DCM direction cosine matrix.

EKF extended Kalman filter.

FOV field of view.

GPS Global Positioning System.

HAV high altitude vehicle.

IMU inertial measurement unit.

INS Inertial Navigation System.

LLH latitude-longitude-height.

LOP line of position.

NED North-East-Down.

PSF point spread function.

ROI region of interest.

RPA remotely piloted aircraft.

RPY roll-pitch-yaw.

RSS root sum square.

UKF unscented Kalman filter.

CELESTIAL AIDED INERTIAL NAVIGATION
BY TRACKING HIGH ALTITUDE VEHICLES

I. Introduction

Celestial navigation has been shown to be a promising alternative navigation method in the absence of the Global Positioning System (GPS) for autonomous navigation [14]. Further research into celestial navigation technology is important for the United States as the Department of Defense continues to seek non-GPS navigation solutions. Celestial Navigation Systems (CNSs) can determine a complete navigation solution (position, velocity, and attitude) with respect to an inertial reference frame. Attitude is determined by observing stars whose positions are known [8] while position and velocity is obtained by observing reference objects with known positions and velocities against a background of stars [6]. Most research involving celestial navigation involve space-based operations to determine only the attitude of spacecraft. Research in CNS for a complete navigation solution within the Earth's atmosphere is sparse. This research focuses on the latter, evaluating the performance of using an Inertial Navigation System (INS) aided by CNS on a remotely piloted aircraft (RPA) to determine its position, velocity, and attitude.

Celestial navigation functions by imaging celestial objects of known positions in an inertial reference frame. After pattern recognition and identification, the CNS can determine where it's pointing in the inertial reference frame and therefore it knows its attitude. Because stars are at such great distances away from the Earth, they are considered to be stationary in the inertial frame and their positions treated as fixed reference points for navigating purposes. As a consequence, the location of the stars

are commonly described in an inertial reference frame in angular coordinates rather than Cartesian coordinates. Directly analogous to the Earth's geodetic longitude (λ) and latitude (ϕ), the angular coordinates of the stars in the inertial frame are given in terms of right ascension (α) and declination (δ) relative to the vernal equinox and the equatorial plane [18]. Due to the long history of astronomy, many of the visible star's coordinates are precisely known. As long as a CNS can identify and recognize stars within its field of view (FOV), it can determine its pointing direction.

Calculating accurate attitude requires tracking the stars at sub-pixel level accuracy. To get sub-pixel accuracy on point sources, star trackers are typically defocused to spread the point source's intensity over multiple pixels instead of focusing all of the light onto one pixel [8]. This spread of intensity over multiple pixels allows the star tracker to get a better estimate of the point source image's center through a process known as centroiding. Once the image center is calculated, the star tracker generates a pointing vector to the point source in the camera's reference frame, which can then be transformed to a more readable navigation solution position vector (i.e., to an inertial, Earth-fixed, or a local navigation reference frame), assuming that the rotation from the camera to the observer's body frame is known. Centroid accuracy directly contributes to the star tracker's overall performance accuracy, commonly described in terms of boresight and cross-boresight accuracy, so it is important to get the centroid as accurate as possible.

Unfortunately, position is unobtainable from imaging stars only because the stars are treated as being an infinite distance away. To solve this problem, the CNS must observe a reference object moving against a background object to update its own position, known as angles-only navigation [6]. The caveat is that the observer must know the positions of both reference and background objects at the time of the observation. Since most CNSs point skyward to track the stars and the stars' positions

are known precisely, the stars make for great background objects. After observing the reference object in its FOV against a starry background, the CNS can triangulate its position and velocity [6]. In this research, a secondary aircraft, referred to a high altitude vehicle (HAV), will be used as the reference object moving against a background field of stars to determine the RPAs position and velocity.

Research using celestial navigation has been dominated by space-born applications, as star trackers are widely used for attitude determination on spacecrafts. However in space-based missions, position is commonly determined by Earth horizon sensing. This is possible at the orbital altitudes of spacecraft. Ning and Fang used both direct and indirect horizon sensing combined with CNS on low Earth orbit satellites and compared the results of fusing the measurements with an extended Kalman filter (EKF) versus an unscented Kalman filter (UKF) [12]. Their results showed position accuracies of <150 m and <70 m with the EKF and the UKF, respectively. In contrast, this research simulates a trajectory near the Earth's surface and used the CNS to triangulate the navigating vehicle's position and velocity by observing the reference HAV.

Some work has been done with celestial navigation for ballistic missile or launch vehicle navigation, which is more relatable to this research than spacecraft navigation. However the altitudes reached are still very high and horizon sensors may still be used. Rad et al. combined INS, CNS, and horizon sensing for ballistic missile navigation, resulting in altitude errors of <300 m [16]. However due to the trajectory of the flight, the missile changed its sensor configurations at different altitude phases. The CNS stayed on the entire time but the INS and horizon sensors were altitude dependent. In this research, the RPA flies at a constant altitude and does not change sensor configurations mid-flight.

Ali and Fang used an unscented particle filter to combine INS and CNS measure-

ments for ballistic missile navigation, but relied solely on inertial measurement unit (IMU) integration to get position and velocity measurements [1]. While they were able to reduce the errors by introducing a new axes misalignment error modeling technique, the navigation solution was still subject to drift with an attitude error of >600 m after a 30 min simulation. Similarly, Nobahari et al. integrated IMU measurements to get position and velocity updates with CNS for attitude aiding only, but they used back-propagation and smoothing techniques with a UKF which stopped the position error drift [13]. However as in [16], both [1] and [13] also changed sensor configurations based on flight altitudes. In their research, the CNS was the altitude-dependent sensor, turning on only at high altitudes. The sensors used in this research are altitude independent and does not require advanced filtering techniques like back-propagation or smoothing.

While directly similar research is sparse, there has been some work with CNS aiding on aircraft or terrestrial navigation. Alkhaldi simulated a flight with GPS, INS, and CNS sensors, combining the measurements with a Linear Kalman Filter [2]. The GPS sensor was cut time time after the simulation began and relied on INS and CNS for the rest of the flight. Alkhaldi simulated various grade INS and showed the position and tilt error covariance with and without CNS. The commercial grade INS had tremendous improvements in navigation performance with the benefit of the CNS while the navigation grade INS had marginal improvements. However regardless of the INS grade, the position errors were still subject to grow without bound due to drift due to not having direct position updates. This research will have position, velocity, and attitude measurement updates with the CNS using the angles-only triangulation algorithm.

In his dissertation, Pierce developed an EKF model of a stationary CNS-INS system, which includes the CNS angles-only triangulation algorithm by tracking satellites

as the reference object [15]. His model allowed various tunable parameters such as the grade of IMU, the orbital altitudes of the satellites imaged, the frequency between observations, etc. Diaz extended Pierce's work in his thesis to extensively evaluate differential ephemeris correction within Pierce's model [3]. Diaz modified the distance between the remote and reference observation sites as well as adding a time delay to the correction and showed its effects on the navigation solution. This research is similar to the work Pierce and Diaz performed; however, instead of a stationary model fixed on the ground, this research uses a mobile observer taking measurements of an HAV instead of satellites.

A MATLAB simulation tool, Celestial Aided Inertial Navigation Subsystem (CAINS), that was provided by the sponsor, Air Force Research Laboratory, was used for all simulations performed. Several simulations of an RPA equipped with an INS, CNS, and barometer were run to determine the navigation accuracy in multiple scenarios. Within the tool, the sensor measurements are combined using an EKF to generate a prediction based on the measurements available. This thesis is divided into five chapters. Chapter II covers a background of the material, including an in-depth summary of the CNS absolute triangulation algorithm as well as a generic description of EKF functionality. Chapter III walks through the research methodology, providing a detailed description of the CAINS algorithm and the specific linearized models for the EKF. Chapter IV presents an analysis of the results. Chapter V concludes the paper with a summary of the research performed and potential future research.

II. Background

This chapter includes background material on star trackers, CNS operation, and Kalman filtering. Section 2.1 define the different frames of reference that will be discussed in this paper. Section 2.2 describes typical star tracking attitude determination and accuracy. Section 2.3 explains how to acquire position and velocity from a CNS using absolute triangulation and Section 2.4 shows how to ensure the reference object used by the triangulation algorithm is detectable by the star tracker. Section 2.5 closes this chapter with an overview of Kalman filters, specifically the EKF.

2.1 Reference Frames

Before describing CNS operations in detail, it is necessary to define the different frames of reference. Some reference frames are time dependent, such as inertial frames. This paper will assume the J2000 epoch for such reference frames. J2000 is defined as the epoch on the Julian date 1 January 2000, 12:000 terrestrial time [19].

- The Earth-centered inertial reference frame (i-frame) is a non-rotating terrestrial frame of reference whose origin is at the Earth's center [19]. The i_i- and j_i-axes lie on the equatorial plane, where the i_i-axis points to the vernal equinox and the j_i-axis points 90° East. The k_i-axis points orthogonal to the equatorial plane towards the North pole. See Figure 1.

- The Earth-centered Earth-fixed reference frame (e-frame) is a terrestrial frame of reference whose origin is also at the Earth's center and the axes lie on the same planes as the i-frame (along the equatorial planes and towards the pole). However the e-frame is a rotating frame of reference that moves with the Earth's rotation [19]. The i_e-axis points to the intersection of the prime meridian and

the equator, the j_e-axis points 90° East and the k_e-axis points to the North pole. See Figure 1.

- The Geodetic reference frame (g-frame) is a terrestrial frame of reference with respect to the reference ellipsoid [18]. The g-frame uses latitude-longitude-height (LLH) instead of absolute meters from the center of the Earth. The latitude ϕ, measures the North-South angle from the equator while the longitude λ, measures the East-West angle from the prime meridian, and the height measures the distance from the surface of the ellipsoid at a given ϕ and λ. This paper will use the World Geodetic System 84 as the reference ellipsoid. The g-frame is also referred to as the LLH frame. See Figure 1.

- The navigation frame (n-frame) is a frame of reference with respect to the observer, whose origin is the center of the observer [19]. The x_n- and y_n-axes lie on the plane of the horizon, where the x_n-axis points North and the y_n-axis points East. The z_n-axis points down, orthogonal to the plane of horizon towards the Earth's center. This frame is also referred to as the North-East-Down (NED) frame. See Figure 1.

- The body frame (b-frame) is a frame of reference with respect to the observer, whose origin is the center of the observer. However, unlike the n-frame whose axes are fixed, the b-frame rotates with the observer's attitude [19]. The x_b-axis points in the observer's facing direction, the y_b-axis points 90° to the observer's right, and the z_b-axis points 90° below the observer. This frame is also referred to as the roll-pitch-yaw (RPY) frame, defining roll as a tilt about the x_b-axis, pitch about the y_b-axis, and yaw about the z_b-axis.

- The camera frame (c-frame) is similar to the b-frame in that it is a frame of reference with respect to the camera [19]. The origin is at the camera's center

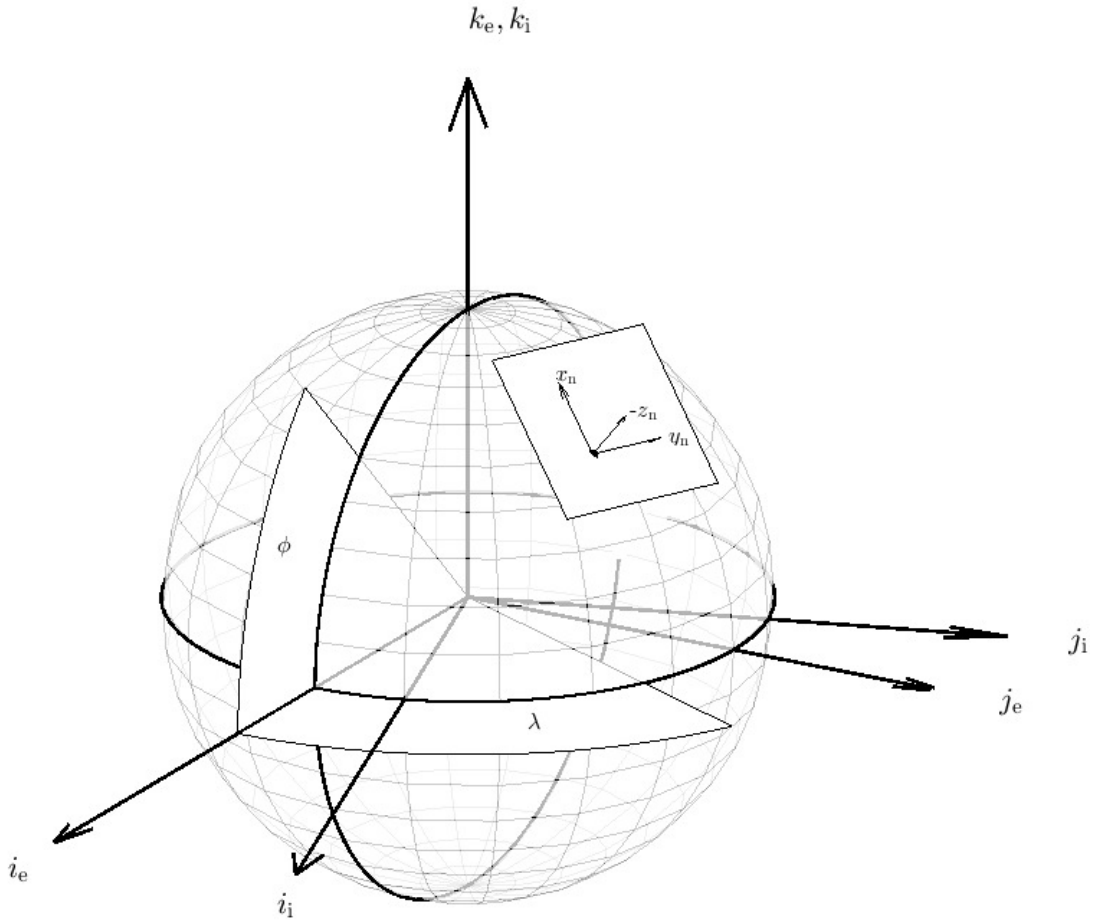


Figure 1. Coordinate frames, showing the *i*-frame, *e*-frame, *g*-frame and *n*-frame. The i_i-axis always points to the vernal equinox while the i_e-axis always points to the intersection of the equator and the prime meridian (highlighted). ϕ and λ show the latitude and longitude, respectively, of the local *n*-frame origin. Note that the *n*-frame shows a *negative* z_n-axis pointing up from the local horizon for graphic purposes only.

and the axes move with the camera's attitude. However, unlike the b-frame, z_c-axis points towards the direction the camera is facing, or the optical axis. The x_c-axis points to the top of the camera (aligned with the vertical pixels) and the y_c-axis points to the right of the camera (aligned with the horizontal pixels).

- The pixel frame (p-frame) is a 2D frame of reference describing the location of an image on the camera's focal plane [5]. The origin of the p-frame is aligned with the camera's optical axis at the center of the focal plane.

In this paper, vectors in a specific frame of reference have a superscript indicating the appropriate frame, and a direction cosine matrix (DCM) transforming one frame to another are formatted such that the original frame is indicated by a subscript and the resulting frame is indicated by a superscript. For example, a position vector in the e-frame will appear as $\mathbf{x}^e$, and a DCM from the e-frame to the n-frame will appear as $\mathbf{C}_e^n$.

2.2 Star Tracker Operation: Attitude Acquisition

The operation of a CNS begins by taking an image of the sky using a star tracking camera. Stars appear as point sources and produce 2D Gaussian impulse responses through optical systems known as a point spread function (PSF) [5]. Stars are generally very dim so a detection threshold must be set to separate the PSF from background noise. Because an image of the sky is mostly empty, processing is performed within an $n \times n$ pixel window around each potential object location, known as the region of interest (ROI). The detection limit is calculated from the raw image data, $d(x^p, y^p)$, at the p-frame coordinates (x^p, y^p), as a background removed signal to noise ratio normalized by the noise standard deviation [20],

$$S/N(x^p, y^p) = \frac{d(x^p, y^p) - B}{\sigma} \tag{1}$$

where B is the background noise value and σ is the standard deviation of B. The background noise can be obtained by taking the median value of $d(x^p, y^p)$ within an empty ROI, and the standard deviation is calculated by

$$\sigma = \sqrt{\frac{\sum_{w=1}^{n}\sum_{z=1}^{n} d(w,z)^2}{n^2} - B^2} \tag{2}$$

where w and z are the ROI pixel locations and n is the number of pixels across the ROI. By setting a detection limit, γ, which represents the number of standard deviations above the background noise, then an object is detected if the signal to noise ratio is greater than γ.

Once a star is detected, each star's center must be precisely determined. Star centroiding allows the center to be calculated with sub-pixel accuracy. The centroid location is calculated from the background-removed image within the ROI,

$$\begin{aligned} x^p_{\text{cen}} &= \sum_{w=1}^{n}\sum_{z=1}^{n} \frac{w\ (d(w,z)-B)}{DN} \\ y^p_{\text{cen}} &= \sum_{w=1}^{n}\sum_{z=1}^{n} \frac{z\ (d(w,z)-B)}{DN} \end{aligned} \tag{3}$$

where DN is the brightness of the background-removed ROI,

$$DN = \sum_{w=1}^{n}\sum_{z=1}^{n} (d(w,z)-B). \tag{4}$$

The result from Equation (3) gives the centroid position in the p-frame [8], which is then transformed to a unit pointing vector in the c-frame by

$$\begin{bmatrix} x^c \\ y^c \\ z^c \end{bmatrix} = \begin{bmatrix} \cos\left(\arctan\left\{\dfrac{\Delta x}{\Delta y}\right\}\right)\cos\left(\dfrac{\pi}{2}-\arctan\left\{\sqrt{\left(\dfrac{\Delta x}{f}\right)^2+\left(\dfrac{\Delta y}{f}\right)^2}\right\}\right) \\ \sin\left(\arctan\left\{\dfrac{\Delta x}{\Delta y}\right\}\right)\cos\left(\dfrac{\pi}{2}-\arctan\left\{[\sqrt{\left(\dfrac{\Delta x}{f}\right)^2+\left(\dfrac{\Delta y}{f}\right)^2}\right\}\right) \\ \sin\left(\dfrac{\pi}{2}-\arctan\left\{\sqrt{\left(\dfrac{\Delta x}{f}\right)^2+\left(\dfrac{\Delta y}{f}\right)^2}\right\}\right) \end{bmatrix} \tag{5}$$

where Δx and Δy are the distances of the centroid coordinate from the optical axis coordinate, $(\Delta x, \Delta y) = (x^p_{\text{cen}} - x^p_o, y^p_{\text{cen}} - y^p_o)$, and f is the focal length. Figure 2 shows a diagram of a simple pinhole camera model, highlighting the ROI and the

unit vector pointing to the star in the c-frame.

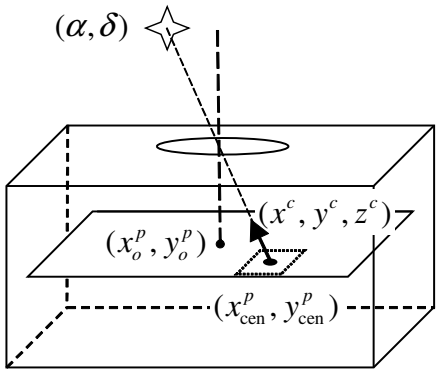


Figure 2. Simple Camera Diagram. The star's position is given in the i-frame in terms of right ascension and declination. The star's image is processed around a small ROI to determine its centroid coordinates in the p-frame, leading to the unit pointing vector in the c-frame.

It is necessary to convert the c-frame pointing vector to the appropriate frame of reference to perform star matching. Star catalogs record star coordinates in inertial reference frames in terms of angles of right ascension and declination, (α, δ), pointing to their location [18]. Various methods exist to transform a vector from the c-frame unit vector to the appropriate inertial frame [16]. Regardless of the method used, at least three stars must be visible in the FOV for star matching and attitude determination [4]. The conversion from a rectangular vector, (i, j, k), to (α, δ) and back from (α, δ) to (i, j, k) is given by Equations (6) and (7), [18]

$$\begin{bmatrix} \alpha \\ \delta \end{bmatrix} = \begin{bmatrix} \arctan\left(\dfrac{j}{i}\right) \\ \arctan\left(\dfrac{k}{\sqrt{i^2 + j^2}}\right) \end{bmatrix} \tag{6}$$

$$\begin{bmatrix} i \\ j \\ k \end{bmatrix} = \begin{bmatrix} \cos(\delta)\cos(\alpha) \\ \cos(\delta)\sin(\alpha) \\ \sin(\delta) \end{bmatrix}. \tag{7}$$

After measuring the attitude, it's necessary to determine the accuracy of that

measurement. Star tracker accuracy is commonly described by its boresight accuracy (z-axis) and cross-boresight (x- and y-axis) accuracy, and is directly affected by the accuracy of the centroid calculation from Equation (3). Given an average centroiding accuracy, E_{cen}, the camera's boresight and cross-boresight accuracy can be calculated as [8]

$$E_{\text{bore}} = \arctan\left(\frac{E_{\text{cen}}}{0.3826 N_{\text{pixel}}}\right)\frac{1}{\sqrt{N_{\text{star}}}} \tag{8}$$

$$E_{\text{cross}} = \frac{\theta_{\text{FOV}} E_{\text{cen}}}{N_{\text{pixel}} N_{\text{star}}} \tag{9}$$

where N_{pixel} is the total number of pixels across the focal plane in the respective axis, N_{star} is the average number of stars detected within the FOV, and θ_{FOV} is the FOV angle in degrees. The value of 0.3826 in Equation (8) comes from the average distance from the optical axis to the imaged PSF, assuming a square $N_{\text{pixel}} \times N_{\text{pixel}}$ focal plane[1]. Generally, E_{bore} is about 10 times less accurate than E_{cross} [8].

2.3 Celestial Navigation for Position and Velocity

Using a star tracker to track just the stars stars solves for the observer's attitude only. To get position or velocity data, observations of foreground and background elements with known positions must be taken, allowing the observer to triangulate its position and velocity. Because the stars have known positions and are readily available for observation, star trackers can be used in this application to observe any foreground reference against background stars, a concept introduced by Kaplan [6]. This section will review his absolute triangulation method.

For a stationary observer and stationary references, the observer's position, $\mathbf{x}$, can be determined by

$$\mathbf{x} = \mathbf{P} + r\mathbf{d} \tag{10}$$

[1] $\int_{-N/2}^{N/2}\int_{-N/2}^{N/2}\sqrt{x^2+y^2}dxdy \approx 0.3826N$

where $\mathbf{P}$ is the reference object's position, $\mathbf{d}$ is a measurement in the form of a unit pointing vector from the observer to the reference object, and r is an arbitrary scaling value. Due to the arbitrary value of r, Equation (10) results in a line of position (LOP) in 3D space, going from the observer to the background object through the reference object, on which the observer's position can exist. To resolve this ambiguity, measurements from at least two references must be taken such that the observer's position narrows down to the intersection of the LOPs. Given n position and measurement vectors, the solution to the stationary observer's position is [6]

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} n - [d_{i_1}^2] & -[d_{i_1} d_{i_2}] & -[d_{i_1} d_{i_3}] \\ -[d_{i_1} d_{i_2}] & n - [d_{i_2}^2] & -[d_{i_2} d_{i_3}] \\ -[d_{i_1} d_{i_3}] & -[d_{i_2} d_{i_3}] & n - [d_{i_3}^2] \end{pmatrix}^{-1} \begin{pmatrix} [P_{i_1} - (\mathbf{d}_i \cdot \mathbf{P}_i) d_{i_1}] \\ [P_{i_2} - (\mathbf{d}_i \cdot \mathbf{P}_i) d_{i_2}] \\ [P_{i_3} - (\mathbf{d}_i \cdot \mathbf{P}_i) d_{i_3}] \end{pmatrix} \tag{11}$$

where the bracketed terms $[\dots]$ represent the summation over the n measurements, $\sum_{i=1}^{n} [\dots]$, while $(P_{i_1}, P_{i_2}, P_{i_3})$ and $(d_{i_1}, d_{i_2}, d_{i_3})$ are the ith $\mathbf{P}_i$ and $\mathbf{d}_i$ vector elements. Equation (11) is a system of three equations and three unknowns, so the observer's position can be calculated exactly.

This algorithm can also be used for a mobile observer. Given an initial position $\mathbf{x}_0$ and initial velocity $\mathbf{v}_0$, the observer's position over time is modeled as

$$\mathbf{x}_i = \mathbf{x}_0 \left(1 + \frac{f_i}{x_0}\right) + \mathbf{v}_0 t_i \tag{12}$$

where $\mathbf{x}_i$ is the observer's position at time t_i, f_i is the curvature of the ellipsoid at t_i (assuming the observer's trajectory is near the Earth's surface), and $x_0 = |\mathbf{x}_0|$ [6]. Substituting Equation (12) into Equation (10) and with n position and measurements

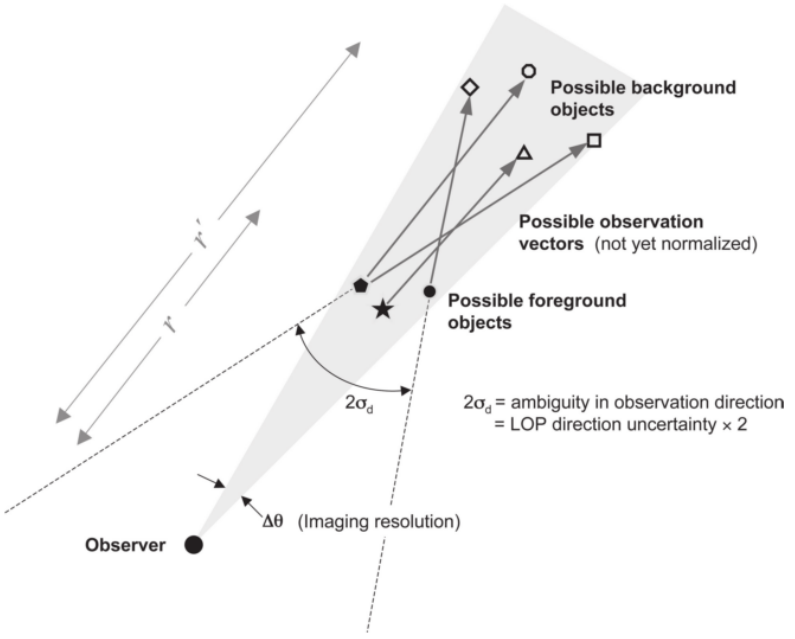
vectors, the solution to the observer's position and velocity is

$$\begin{pmatrix} x_{i_1} \\ x_{i_2} \\ x_{i_3} \\ v_{i_1} \\ v_{i_2} \\ v_{i_3} \end{pmatrix} = \begin{pmatrix} [(d_{i_1}^2-1)\beta_i^2] & [d_{i_1}d_{i_2}\beta_i^2] & [d_{i_1}d_{i_3}\beta_i^2] & [(d_{i_1}^2-1)t_i\beta_i] & [d_{i_1}d_{i_2}t_i\beta_i] & [d_{i_1}d_{i_3}t_i\beta_i] \\ [d_{i_2}d_{i_1}\beta_i^2] & [(d_{i_2}^2-1)\beta_i^2] & [d_{i_2}d_{i_3}\beta_i^2] & [d_{i_2}d_{i_2}t_i\beta_i] & [(d_{i_2}^2-1)t_i\beta_i] & [d_{i_2}d_{i_3}t_i\beta_i] \\ [d_{i_3}d_{i_1}\beta_i^2] & [d_{i_3}d_{i_2}\beta_i^2] & [(d_{i_3}^2-1)\beta_i^2] & [d_{i_3}d_{i_1}t_i\beta_i] & [d_{i_3}d_{i_2}t_i\beta_i] & [(d_{i_3}^2-1)t_i\beta_i] \\ [(d_{i_1}^2-1)t_i\beta_i] & [d_{i_1}d_{i_2}t_i\beta_i] & [d_{i_1}d_{i_3}t_i\beta_i] & [(d_{i_1}^2-1)t_i^2] & [d_{i_1}d_{i_2}t_i^2] & [d_{i_1}d_{i_3}t_i^2] \\ [d_{i_2}d_{i_1}t_i\beta_i] & [(d_{i_2}^2-1)t_i\beta_i] & [d_{i_2}d_{i_3}t_i\beta_i] & [d_{i_2}d_{i_1}t_i^2] & [(d_{i_2}^2-1)t_i^2] & [d_{i_2}d_{i_3}t_i^2] \\ [d_{i_3}d_{i_1}t_i\beta_i] & [d_{i_3}d_{i_2}t_i\beta_i] & [(d_{i_3}^2-1)t_i\beta_i] & [d_{i_3}d_{i_1}t_i^2] & [d_{i_3}d_{i_2}t_i^2] & [(d_{i_3}^2-1)t_i^2] \end{pmatrix}^{-1} \times \begin{pmatrix} [-(P_{i_1}-(\mathbf{d}_i\cdot\mathbf{P}_i)d_{i_1})\beta_i] \\ [-(P_{i_2}-(\mathbf{d}_i\cdot\mathbf{P}_i)d_{i_2})\beta_i] \\ [-(P_{i_3}-(\mathbf{d}_i\cdot\mathbf{P}_i)d_{i_3})\beta_i] \\ [-(P_{i_1}-(\mathbf{d}_i\cdot\mathbf{P}_i)d_{i_1})t_i] \\ [-(P_{i_2}-(\mathbf{d}_i\cdot\mathbf{P}_i)d_{i_2})t_i] \\ [-(P_{i_3}-(\mathbf{d}_i\cdot\mathbf{P}_i)d_{i_3})t_i] \end{pmatrix} \tag{13}$$

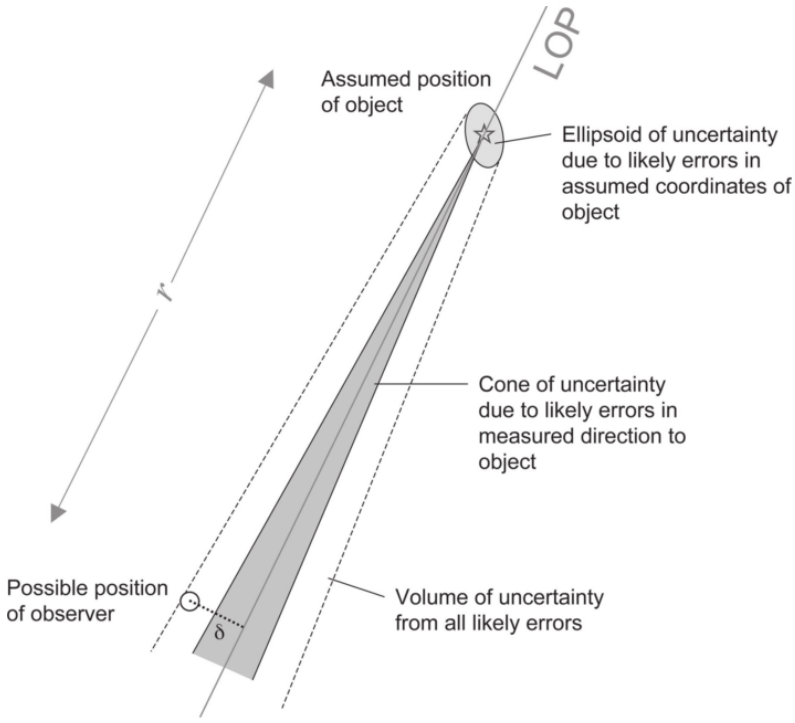
where the bracketed terms represent the same summation as in Equation (11) and β is a shorthand expression for the curvature term, $\beta = \left(1 + \frac{f_i}{x_0}\right)$. The cross symbol $\times$ in Equation (13) represents simple matrix multiplication and not a cross product of matrices. As a point of clarification, while it was previously mentioned that n measurements were necessary, a mobile observer could take n measurements of a single reference object over a period of time rather than requiring n measurements at each timestep.

With perfect predictions and measurements, the observer's position should be exactly on the LOPs. Accounting for errors in both $\mathbf{P}$ and $\mathbf{d}$, however, generates uncertainty around $\mathbf{x}$, so the observer's position will be some distance away from the LOPs. The uncertainty around $\mathbf{x}$ is [6]

$$\sigma_{\mathbf{x}}^2 = \sigma_{\mathbf{P}}^2 + r^2\sigma_{\mathbf{d}}^2 \tag{14}$$



(a)



(b)

Figure 3. Uncertainty with absolute triangulation. Figures reproduced from [6]. (3a) Ambiguity in the direction of the LOPs due to multiple objects within the FOV. (3b) Total volume of uncertainty around x (dotted) due to $\sigma_{\mathbf{P}}$ (ellipsoid) and $\sigma_{\mathbf{d}}$ (cone). The error in observer's position is δ.

where each σ represents the uncertainties associated with their respective vectors, and r is the distance to the reference object. $\sigma_{\mathbf{P}}$ is obtained externally as it is simply the uncertainty around the reference object's coordinate. However, $\sigma_{\mathbf{d}}$ must be calculated. With multiple reference and background objects within the FOV, there can be ambiguity with the pointing direction. Figure 3a shows this uncertainty of the measurement, $\mathbf{d}$. The uncertainty is limited by the camera's imaging resolution, $\Delta\theta$, and can be calculated based on the geometry of the observed objects as

$$\sigma_{\mathbf{d}} = \Delta\theta \left(\frac{1}{2} + \frac{r}{r' - r} \right) \tag{15}$$

where r and r' are the reference and background object distances, respectively. In this case, $r' \to \infty$ because the stars are so distant, so Equation (15) simplifies to

$$\sigma_{\mathbf{d}} = \frac{\Delta\theta}{2}. \tag{16}$$

Plugging Equation(16) into Equation (14) results in the position uncertainty as

$$\sigma_{\mathbf{x}}^2 = \sigma_{\mathbf{P}}^2 + r^2 \left(\frac{\Delta\theta}{2} \right)^2. \tag{17}$$

Thus, given $\sigma_{\mathbf{P}}$ and $\Delta\theta$, the accuracy of the observer's position is known.

With a stationary observer, $\sigma_{\mathbf{x}}$ forms a spherical volume of uncertainty around the intersection of the LOPs. However with a mobile observer, the LOPs don't intersect but rather converge around the observer's trajectory, creating a volume around the whole trajectory. Figure 3b depicts this volume of uncertainty around $\mathbf{x}$. The position error from the true observer's position to each LOP at time t_i

$$\delta_i = |\mathbf{d}_i \times (\mathbf{P}_i - \mathbf{x}_i)| \tag{18}$$

The solution provided by Equation (11) or Equation (13) is a least-squares method that minimizes the sum of squares of the position errors, $\mathcal{D} = \sum_{i=1}^{n} \delta_i^2$.

2.4 Object Size and Brightness

Since star trackers are designed to identify and track point source objects, it is useful to assume that the reference object will also be a point-like object to avoid extraneous image recognition and detection processing. However, reference objects, due to their relative closeness, may not appear as a point source. To ensure that the reference object is seen as a point source, the Rayleigh criterion can determine the camera's minimum resolution.

Given a single point source generates an Airy disk PSF, then two point sources separated by a distance δ_z generates two PSFs offset by a distance δ_i. The Rayleigh criterion states that an imaging system's minimum resolution of two incoherent point sources happens when the peaks of the Airy disks generated by each point source lies on the first zero of the other's disk [5]. This occurs when the PSF separation distance is

$$\delta_i \geq 1.22 \frac{\lambda z_i}{D_a} \tag{19}$$

where λ is the wavelength of light, z_i is the distance from the aperture to the image at the focal plane, and D_a is the aperture diameter. Figure 4 graphically depicts the Rayleigh criterion.

In other words, if δ_i is less than the Rayleigh criterion, then the two point sources are unresolvable and will appear as a single point source by the optical system. While Equation (19) is written in terms of δ_i and z_i, it can also be used to solve for δ_z given z, the distance from the objects to the aperture, due to the geometry of similar triangles.

Another factor to consider is the brightness of the object in order to be detected by the star tracker. The brightness of any celestial object is typically given by its

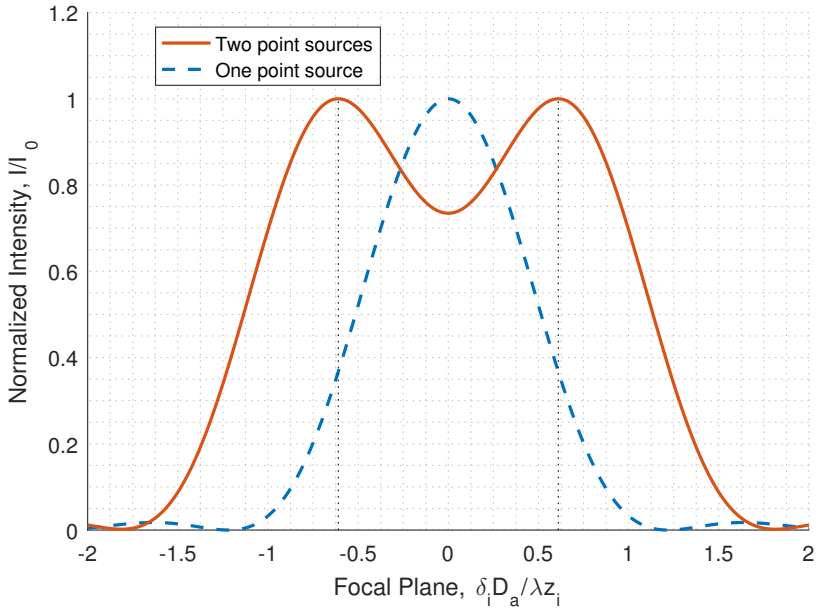


Figure 4. 2D cross sections of an Airy diffraction pattern PSF generated by two point sources. The PSF of one point source is also shown for reference. The minimum resolvable resolution occurs when the PSF peaks lie on each other's first zero, indicated by the vertical dotted lines.

apparent magnitude, which is its brightness compared to a reference point. The apparent magnitude can be calculated as [7]

$$m = -2.5 \log_{10} \left(\frac{L}{L_0} \right) \tag{20}$$

where m is the apparent magnitude of the source object, L is the brightness of the source observed by the camera, and L_0 is the brightness of the reference point. L and L_0, can be determined as the number of photoelectrons that the objects generate within the camera. Treating the object as an ideal black body, the radiated intensity at a given frequency and temperature is [7]

$$I(\nu, T) = \frac{2h\nu^3}{c^2 \left(\exp\left(\frac{h\nu}{kT} \right) - 1 \right)} \tag{21}$$

where ν is the frequency of light, T is the temperature of the source, h is Planck's

constant, c is the speed of light, and k is Boltzmann's constant.

The radiated intensity from Equation (21) is in units of watts per unit area per frequency per steradian. This can be converted to photons per second per frequency by dividing by the energy per photon[2], integrating over the frequencies of the visible spectrum, and multiplying by the surface area of the object, A_s.

$$K = 4\pi \; A_s \int_{\nu_0}^{\nu_1} \frac{I(\nu, T)}{h\nu} d\nu \tag{22}$$

where the 4π factor accounts for the steradians. Equation (22) then describes the total number of photons per second that is generated by the black body model. Given the camera's quantum efficiency, QE, and the integration time, t, the number of photoelectrons generated by the camera from the source object can be derived as [17]

$$L = K \; QE \; t \; \frac{A_{\mathrm{a}}}{A_z} \tag{23}$$

where A_a is the area of the aperture and A_z is the surface area of the propagated light from the object to the camera. Using Vega as the reference point for L_0 and given the star tracker's minimum detectable apparent magnitude, the minimum number of photoelectrons generated by the reference object can be solved for using Equation (20).

2.5 Extended Kalman Filter

This section presents an overview of how the Kalman filtering works. A Kalman filter is an algorithm that estimates a system's states and the state uncertainties over time. It incorporates measurement data when available to make an optimal prediction [9]. The basic Kalman filter is only appropriate for linear models however, so the EKF is used to handle non-linear problems. The key behind the EKF is that it linearizes

[2] $E = h\nu$

the non-linear models at each timestep by the first-order Taylor series approximation [10]. One critical assumption with this linearization is that the higher order terms in the Taylor series are negligible. If this is not true then the EKF is not an appropriate method. The EKF derivation in this section is a general description of the algorithm. The specific functions related to this research is presented in Chapter 3.

For the system state vector, $\mathbf{x}(t)$, and the system input vector, $\mathbf{u}(t)$, the continuous dynamics model is [10]

$$\dot{\mathbf{x}}(t) = \mathbf{f}[\mathbf{x}(t), \mathbf{u}(t), t] + \mathbf{G}(t)\mathbf{w}(t) \tag{24}$$

where $\mathbf{f}[...]$ describes the non-linear system model based on the current state and the input vectors, $\mathbf{w}(t)$ is a zero-mean white Gaussian process noise of strength $\mathbf{Q}(t)$, and $\mathbf{G}(t)$ controls which states are directly affected by the noise $\mathbf{w}(t)$. The noise strength $\mathbf{Q}(t)$ describes the variance of $\mathbf{w}(t)$ and is defined as

$$E\left\{\mathbf{w}(t)\mathbf{w}(t+\tau)^{\mathrm{T}}\right\} = \mathbf{Q}(t)\delta(\tau). \tag{25}$$

The initial state, $\mathbf{x}(t_0)$, is modeled as a Gaussian random vector whose mean and covariance generates the initial state estimate and initial covariance, $\hat{\mathbf{x}}(t_0)$ and $\mathbf{P}(t_0)$.

Along with the state model, the discrete-time measurements are modeled by [10]

$$\mathbf{z}(t_i) = \mathbf{h}[\mathbf{x}(t_i), t_i] + \mathbf{v}(t_i) \tag{26}$$

where $\mathbf{h}[...]$ describes the non-linear observation model and $\mathbf{v}(t_i)$ is a zero-mean white

Gaussian measurement noise of strength $\mathbf{R}(t_i)$ defined as

$$E\left\{\mathbf{v}(t_i)\mathbf{v}(t_j)^{\mathrm{T}}\right\} = \begin{cases} \mathbf{R}(t_i) & t_i = t_j \\ 0 & \text{otherwise} \end{cases}. \tag{27}$$

After a measurement $\mathbf{z}(t_i)$ arrives, the measurement residual is calculated as the difference between the true measurement and the expected measurement, $\hat{\mathbf{z}}(t_i)$, along with the residual covariance $\mathbf{S}(t_i)$ [9],

$$\begin{aligned} \mathbf{r}(t_i) &= \mathbf{z}(t_i) - \hat{\mathbf{z}}(t_i) \\ &= \mathbf{z}(t_i) - \mathbf{h}[\hat{\mathbf{x}}(t_i^-), t_i] \end{aligned} \tag{28}$$

$$\mathbf{S}(t_i) = \mathbf{H}(t_i)\mathbf{P}(t_i^-)\mathbf{H}^{\mathrm{T}}(t_i) + \mathbf{R}(t_i) \tag{29}$$

where $\mathbf{P}(t_i^-)$ is the covariance of the current state estimates[3].

With the residual covariance, the filter gain is computed as [10]

$$\mathbf{K}(t_i) = \mathbf{P}(t_i^-)\mathbf{H}^{\mathrm{T}}(t_i)\mathbf{S}(t_i)^{-1} \tag{30}$$

where $\mathbf{H}(t_i)$ is the Jacobian of the observation model,

$$\mathbf{H}(t_i) = \left.\frac{\partial \mathbf{h}[\mathbf{x}, t_i]}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}(t_i^-)}. \tag{31}$$

The filter gain is used to update the current state estimates and covariance by incorporating the measurement,

$$\hat{\mathbf{x}}(t_i^+) = \hat{\mathbf{x}}(t_i^-) + \mathbf{K}(t_i)\mathbf{r}(t_i) \tag{32}$$

[3] t_i^- represents the state estimates at time t_i before incorporating the measurement at time t_i, and t_i^+ represents the state estimates after incorporating the measurement.

$$\mathbf{P}(t_i^+) = \mathbf{P}(t_i^-) - \mathbf{K}(t_i)\mathbf{H}(t_i)\mathbf{P}(t_i^-) \tag{33}$$

Finally, the updated states and covariance are propagated forward by integrating [10]

$$\dot{\hat{\mathbf{x}}}(t_{i+1}) = \mathbf{f}[\hat{\mathbf{x}}(t_i^+), u(t), t] \tag{34}$$

$$\dot{\mathbf{P}}(t_{i+1}) = \mathbf{F}(t)\mathbf{P}(t_i^+) + \mathbf{P}(t_i^+)\mathbf{F}^{\mathrm{T}}(t) + \mathbf{G}(t)\mathbf{Q}(t)\mathbf{G}^{\mathrm{T}}(t) \tag{35}$$

where $\mathbf{F}(t)$ is the Jacobian of the state model,

$$\mathbf{F}(t) = \left.\frac{\partial \mathbf{f}[\mathbf{x}, \mathbf{u}(t), t]}{\partial \mathbf{x}}\right|_{\mathbf{x}=\hat{\mathbf{x}}(t_i^+)}. \tag{36}$$

The solutions to Equations (34) and (35) can be written as

$$\hat{\mathbf{x}}(t_{i+1}^-) = \hat{\mathbf{x}}(t_i^+) + \int_{t_i}^{t_{i+1}} \mathbf{f}[\hat{\mathbf{x}}(t_i^+), \mathbf{u}(t), t]dt \tag{37}$$

$$\begin{aligned}\mathbf{P}(t_{i+1}^-) =& \mathbf{\Phi}[t_{i+1}, t_i; \hat{\mathbf{x}}(t_i^+)]\mathbf{P}(t_i^+)\mathbf{\Phi}^{\mathrm{T}}[t_{i+1}, t_i; \hat{\mathbf{x}}(t_i^+)] \\ &+ \int_{t_i}^{t_{i+1}} \mathbf{\Phi}[t_{i+1}, t_i; \hat{\mathbf{x}}(t_i^+)]\mathbf{G}(t)\mathbf{Q}(t)\mathbf{G}^{\mathrm{T}}(t)\mathbf{\Phi}^{\mathrm{T}}[t_{i+1}, t_i; \hat{\mathbf{x}}(t_i^+)]dt\end{aligned} \tag{38}$$

where $\mathbf{\Phi}(t_{i+1})$ is the state transition matrix calculated from the linearized state model $\mathbf{F}(t)$ [9],

$$\mathbf{\Phi}[t_{i+1}, t_i; \hat{\mathbf{x}}(t_i^+)] = e^{\mathbf{F}(t)\Delta t}. \tag{39}$$

These propagated state estimates then become the current state estimates, or $\hat{\mathbf{x}}(t_{i+1}^-) = \hat{\mathbf{x}}(t_i^-)$, as the EKF algorithm continues to the next iteration.

2.6 Chapter II Summary

This chapter covered the basic information necessary to understand how star trackers are used for position, velocity, and attitude. A basis for the various coordinate frames used in this paper was defined in Section 2.1. Normal star tracker attitude acquisition was described in Section 2.2 as well as position and velocity triangulation in Section 2.3. Section (2.4) explained the necessary conditions required for the reference to be detected. Finally Section 2.5 described the process of a generic EKF. The next chapter will walk through the specific simulation description.

III. Methodology

This chapter describes the CAINS simulation tool which was used to generate all the results in Chapter IV. Section 3.1 provides an overview of the tool itself and a description of the scenario that the tool simulates. Some key assumptions are discussed in Section (3.2). Section 3.3 steps through the CAINS algorithm, describing how measurements are read and processed. Section 3.4 describes the sensor parameters and the linearized **F** and **H** models used by the EKF. Finally Section 3.5 describes how new measurement data was generated.

3.1 CAINS Description

CAINS is a robust simulation tool developed in the MATLAB environment for INS trade studies. It can combine measurements from various sensors into a Kalman filter to generate a navigation solution. The tool simulates an RPA with an INS and other sensors attempting to determine it's navigation solution. The INS/IMU sensor is always on while additional sensors can be toggled on for each new simulation run. The suite of sensors include CNS, GPS, magnometer, and altimeter, amongst other sensor modules.

When a sensor is selected for a simulation run, the tool generates a structure for each sensor that contains all the necessary data required for the sensor to interact with the simulation, such as the sensor-specific parameters, the truth and measurement file locations, and common function handles. Having common function handles is what gives the tool the flexibility to work with multiple sensors, allowing the user to toggle any number of sensors without changing the simulation's inner code. In this research, only the altimeter and the star tracker were used as additional sensors. While the altimeter aids the vertical measurement, the primary focus will be on the star tracker

sensor.

The simulation models a 10 minute flight of an RPA flying directly North-West with an initial geodetic position of $(25°, -90°, 5486.4\text{ m})$. The RPA maintains an average velocity of 152 m/s and though it flies mostly level, it experiences attitude fluctuations most heavily in the roll and pitch axes. The RPA has two fixed star tracking sensors pointing directly up from the body with an 8° FOV. While both star sensors have similar characteristics and are pointing in the same direction, one is used to observe only the stars while the other is used to observe the HAV. In this paper, to differentiate the two sensors one will be called the star tracker and the other will be called the HAV tracker. The HAV follows a similar trajectory and velocity as the RPA but flies at an average altitude of 20 km. With this setup, three different scenarios were run using CAINS:

1. Processing only star measurements for attitude updates.
2. Processing only HAV measurements for position and velocity updates.
3. Combining both star and HAV measurements for position, velocity, and attitude updates.

Based on the number of sensors in use, the size of the state vector is dynamically generated. At a minimum, there are 21 states generated by the INS and IMU alone. The INS generates 9 states: LLH position, NED velocity, and RPY attitude. The IMU generates 12 states: accelerometer and gyro bias states plus accelerometer and gyro scale factors in each of vehicle's three rotation axes. In addition to the minimum 21 states, each star tracking sensor adds two states: the azimuth and elevation bias for that sensor. The altimeter was modeled as unbiased but had white noise added into the measurements so it does not add any states to the state vector. State data is stored in a another structure that saves the state values every time the states are

updated via the Kalman filter's propagation or measurement update. The order of the states in the state vector generated by CAINS is shown in Equation (40), where $p_{\phi\lambda h}$ are the geodetic positions for latitude, longitude, and altitude, $\mathbf{v}_{\text{NED}}$ is the NED velocity vector, $\boldsymbol{\psi}$ is the tilt vector, $\mathbf{b}_a$ and $\mathbf{b}_g$ are the IMU accelerometer and gyro bias vectors, respectively, along the three rotation axes, $\mathbf{sf}_a$ and $\mathbf{sf}_g$ are the IMU accelerometer and gyro scale factor vectors, respectively, along the three rotation axes, and $\mathbf{b}_{\text{ST}}$ and $\mathbf{b}_{\text{HT}}$ are the azimuth and elevation bias vectors for the star tracker and HAV tracker, respectively.

$$\mathbf{x}(t_i) = \begin{bmatrix} p_\phi & p_\lambda & \mathbf{v}_{\text{NED}} & \boldsymbol{\psi} & p_h & \mathbf{b}_a & \mathbf{b}_g & \mathbf{sf}_a & \mathbf{sf}_g & \mathbf{b}_{\text{ST}} & \mathbf{b}_{\text{HT}} \end{bmatrix} \tag{40}$$

It should be mentioned that CAINS can do more than described in this paper. However the full functionality of the tool goes beyond the scope of this research so the algorithm's description has been limited to it's use here.

3.2 Algorithm Assumptions

There are some key assumptions that the model makes:

1. Atmospheric effects are ignored for a simplistic model.
2. Communication between the RPA and the HAV is instant.
3. Image processing and pattern recognition for stellar measurements is already performed.
4. The HAV will appear as a point source to the RPAs star sensors.

The first and second assumptions are made to simplify the model, as this algorithm was made to test the concept of HAV tracking. The third assumption is made to limit the scope of this the research on the estimation aspect, as image processing of

CNS can be its own research area by itself. However, while CAINS doesn't simulate the imaging aspect, it still generates the navigation state updates based on the input measurements. The CNS measurement input only provides the location or the pointing vector to the stellar object; CAINS then processes that data to generate position, velocity, and attitude updates. The fourth assumption however can be validated using optics theory. While the plane itself is unlikely to appear as a point source, it can be assumed that the plane carries a beacon for the HAV tracker to track. From the Rayleigh criterion described by Equation (19) in Section (2.4), using an average wavelength of $\bar{\lambda} = 500$ nm, $z = 15$ km and $D_a = 7$ cm, the beacon's diameter must be less than 13.07 cm to appear as a point source by the camera.

The minimum brightness necessary for the camera to detect the beacon can also be determined. As described in Section (2.4), the minimum number of photoelectrons that is generated by the beacon, L, can be determined using Equation (20). Then the beacon's total number of photons per second, K, can be calculated from Equation (23). This requires knowing A_z, the surface area of the beam propagating from the beacon to the camera. Modeling the propagation of light as a cone with a beam divergence of $\theta = 60°$, A_z will be the area of a circle of radius $r = z\tan(\theta)$. Finally, the minimum wattage required by the beacon is calculated as the product of K and the energy per photon,

$$W = K\ h\nu. \tag{41}$$

Due to the second assumption that the CAINS simulation skips the image processing steps of acquiring its measurements, values for quantum efficiency and integration time were unavailable and assumed as $\overline{QE} = 80\%$ as the average QE over the visible spectrum and $t = 0.01$ s. Given the HAV tracker's minimum detectable magnitude of $m = 6$ and with previous values of z, $\bar{\lambda}$, and D_a, the estimated minimum power required by the beacon is 0.61 W, assuming 100% luminous efficiency. Even with

realistic values for luminous efficiencies, the minimum wattage would still be fairly small. Since the the maximum beacon size and the minimum beacon wattage are reasonably small figures, the last assumption is valid.

3.3 CAINS Algorithm

The process flow for CAINS is described by Figures 5-7. Figure 5 outlines the overall simulation loop while Figure 6 and Figure 7 show the Kalman filter's specific navigation and measurement updates. The INS and star tracker parameters that were used in this simulation are listed in Table 1, where the subscripts a, g, b, sf, ST, and HT represent the respective parameters for the accelerometer, gyro, bias, scale factor, star tracker and HAV tracker.

Table 1. Key sensor parameters for the IMU and CNS used by the EKF.

IMU		**CNS**	
σ_a	2.38×10^{-4} m/s/$\sqrt{\text{s}}$	σ_{ST}	6 arcsec
σ_g	5.82×10^{-7} rad/$\sqrt{\text{s}}$	$\sigma_{\text{b}_{\text{ST}}}$	100 arcsec
σ_{b_a}	2.45×10^{-4} m/s	σ_{HT}	8.49 arcsec
σ_{b_g}	1.45×10^{-8} rad/s	$\sigma_{\text{b}_{\text{HT}}}$	10 arcsec
σ_{sf_a}	100 ppm	τ_{ST}	1000 s
σ_{sf_g}	5 ppm	τ_{HT}	∞
$\tau_{\text{b}_a}, \tau_{\text{b}_g}, \tau_{\text{sf}_a}, \tau_{\text{sf}_g}$	1000 s	FOV	8°

After initializing the sensors and states, each sensor reads measurements from their respective data files and stores the next measurement into the sensor's buffer. The measurement data files contain the time of the measurement, *valid time* or T_{valid}, and the measurement itself. The simulation performs a *sensor peek* where it looks at the buffered data and selects the next earliest measurement, *next measurement*, defined as the measurement with the lowest T_{valid}. If the measurement queue is currently

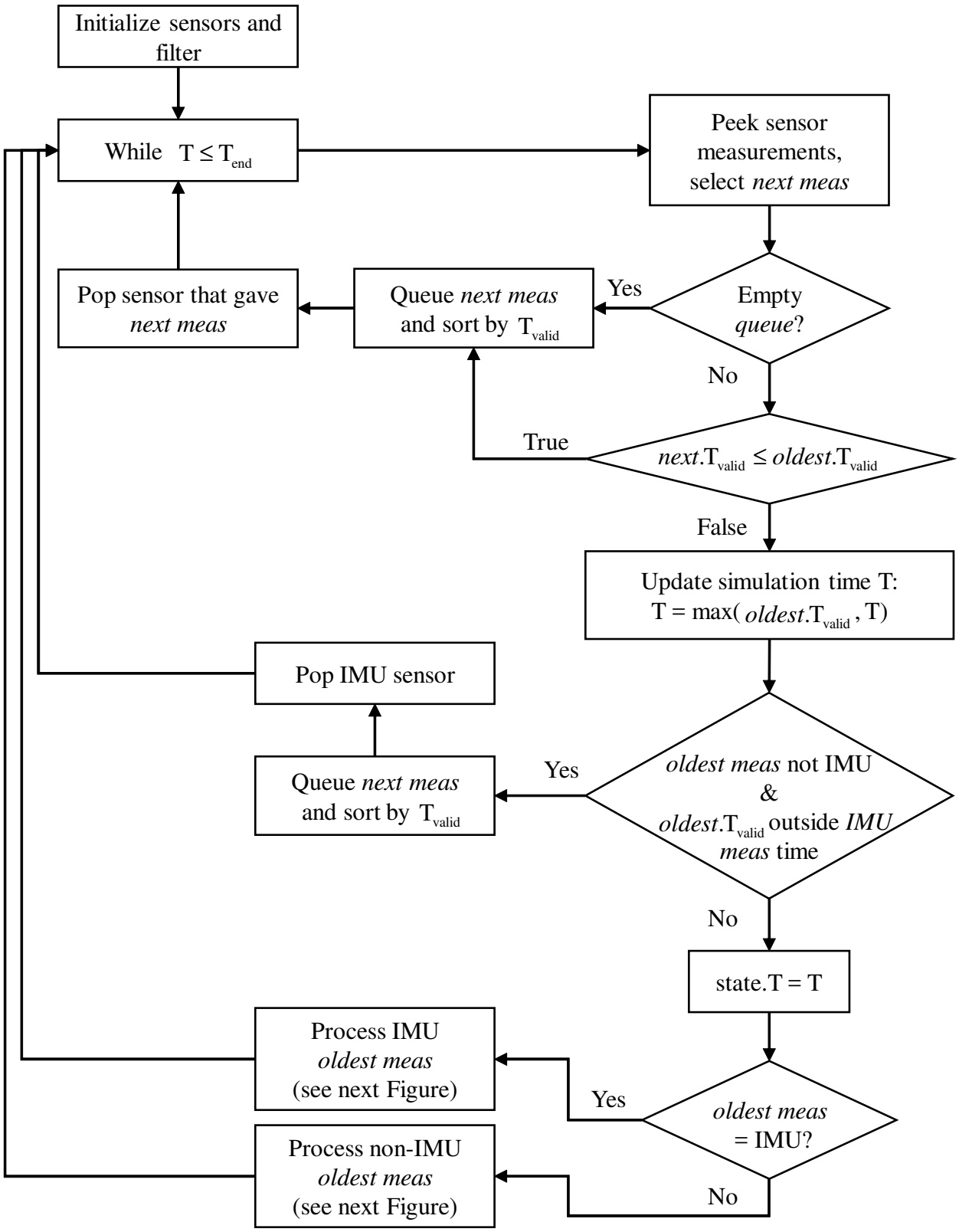


Figure 5. CAINS Process Diagram. See Figures 6 and 7 for more detail.

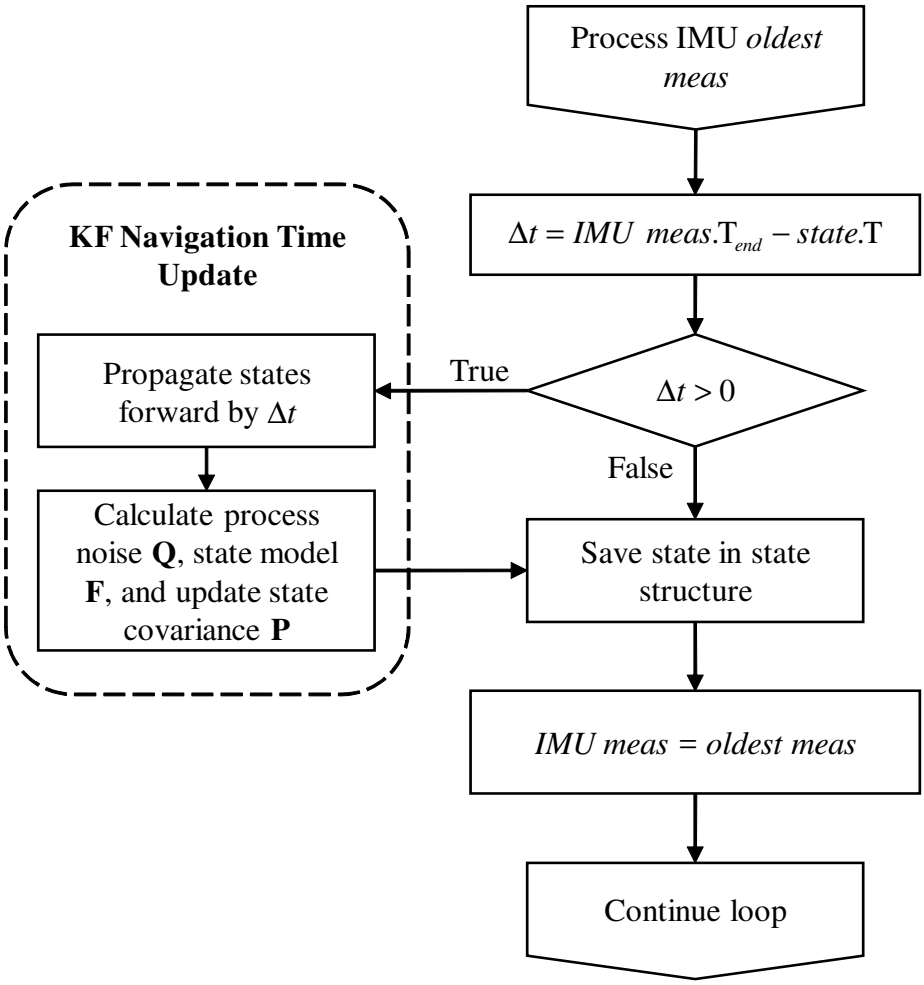


Figure 6. CAINS IMU Measurement Process, expanded from the "Process IMU *oldest meas*" block at the bottom of Figure 5.

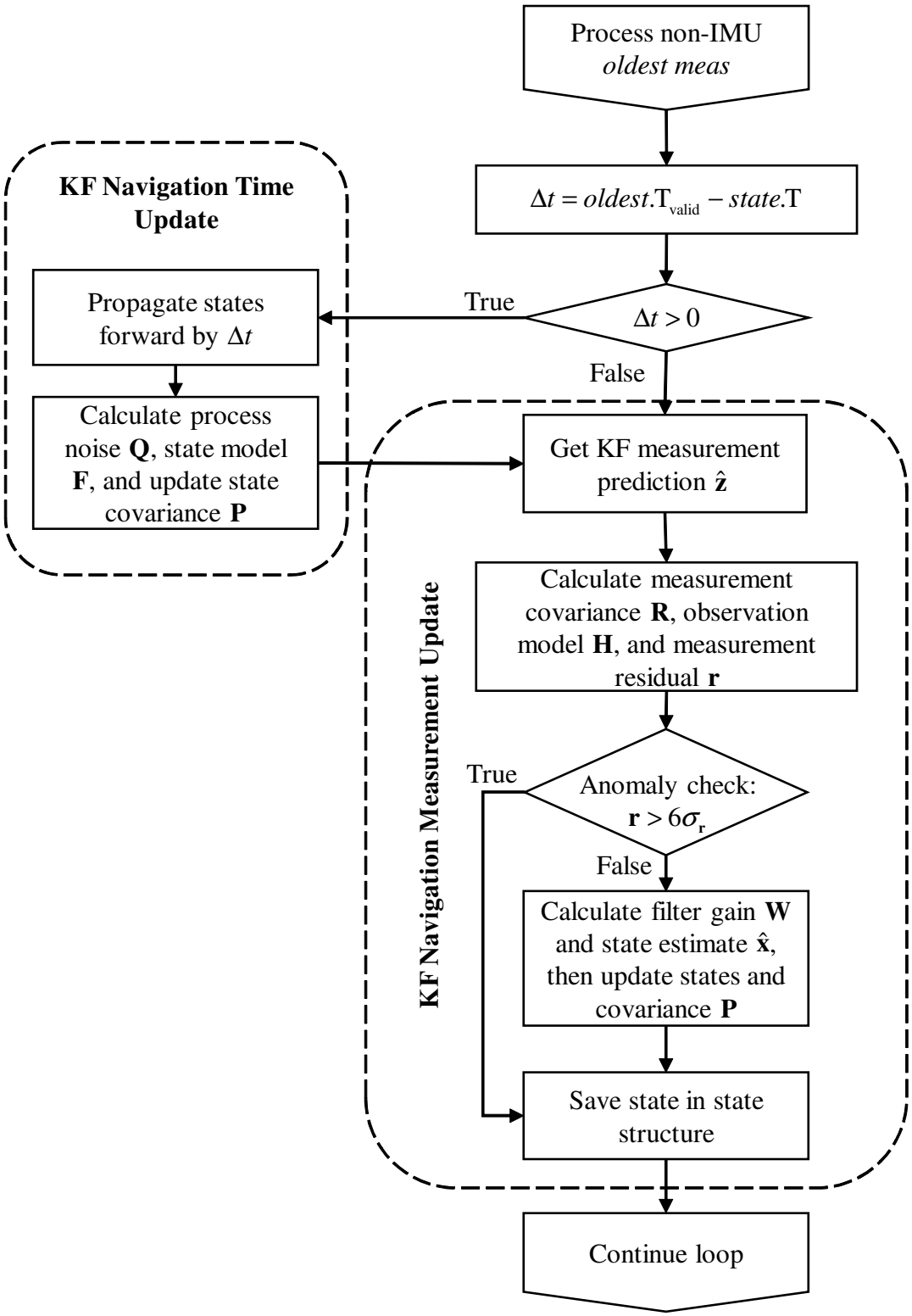


Figure 7. CAINS non-IMU Measurement Process, expanded from the "Process non-IMU *oldest meas*" block at the bottom of Figure 5.

empty then the *next measurement* is stored in the queue. Each time a measurement is added to the queue, the queue is sorted by T_{valid} and the queued measurement with the lowest T_{valid} is marked as the *oldest measurement*. Otherwise if the queue is not empty, then the simulation compares the queue's *oldest measurement's* T_{valid} to the *next measurement's* T_{valid}. The *next measurement* gets queued if it's *valid time* is less than or equal to the *oldest measurement's* time. After queueing, that simulation performs a *sensor pop* in which the simulation loads the next measurement into the sensor buffer for the sensor that provided the *next measurement* that just got queued.

The peek-queue-pop process continues until the *next measurement's* T_{valid} is greater than the *oldest measurement's* T_{valid}. This happens when all the sensor's current measurements are queued and the next set of sensor measurements within the buffer are at future times. The simulation time, T, is updated and the measurement queue is almost ready for processing. There is one other check made before the queued measurements are processed. Because CAINS always has the IMU on, the IMU roughly acts as the simulation's timer. As such, CAINS only processes measurements when the T_{valid} of a non-IMU measurement falls within the current simulation time, given by the current *IMU measurement's* [1] time window. If a non-IMU *oldest measurement* is outside of the *IMU measurement's* time window, then it means the *oldest measurement* is a future measurement that the simulation hasn't caught up to yet, so it queues the *next measurement*, pops the next IMU measurement and continues the loop. The IMU pop acts to progress the IMU forward until a new IMU measurement is assigned as the current *IMU measurement* which captures the *oldest measurement's*

[1]Note that the simulation tracks two IMU measurements: the next measurement from the measurement queue (the *oldest measurement*) which could potentially be from the IMU, and the simulation's current IMU measurement referred to as *IMU measurement*. On the first iteration through the simulation loop, the current *IMU measurement* is initialized with no data so the first *oldest measurement* that comes from the IMU is then assigned as the current *IMU measurement* and the simulation returns to the top of the loop. Once the current *IMU measurement* has been assigned, the algorithm runs normally.

T_{valid}. One key assumption for this algorithm is that the IMU provides measurements at a much greater frequency than the other sensors.

When all the checks are done, the *oldest measurement* is finally ready for processing. First, the state structure saves the current simulation time, T. Then CAINS handles the *oldest measurement* slightly differently depending on whether the *oldest measurement* is an IMU measurement or not. IMU measurements only trigger a Kalman filter time propagation of the states whereas non-IMU measurements do both time propagation and measurement update. Regardless of which path it takes, the algorithm checks if any simulation time actually passed before propagating the states. It calculates a Δt, which is the difference between the *oldest measurement's* time and the current simulation time, T. State propagation does not happen if Δt is zero, which means multiple measurements at the same time are being processed. In this case no changes are made to the states but the states are saved into the state structure anyway. Otherwise, the simulation propagates each state by Δt based on the time constant τ and calculates the linearized dynamics model, $\mathbf{F}$, and the state uncertainty from the updated covariance matrix, $\mathbf{P}$. The updated states and uncertainty values are saved in the state structure and the algorithm returns to the top of the loop.

For non-IMU measurements, the simulation attempts a time propagation the same way as described for IMU measurements, but it also performs a measurement update even if $\Delta t = 0$ and the time propagation step was skipped. The simulation calculates a measurement prediction, the measurement noise covariance, $\mathbf{R}$, and the linearized measurement model, $\mathbf{H}$. With the actual measurement and the measurement prediction, the measurement residual, $\mathbf{v}$, can be calculated. The residual and its covariance, $\mathbf{S}$, as described by Equation (29), is monitored for measurement anomaly detection. A measurement is anomalous if the residual is greater than $n \times \sigma$, where σ is the

standard deviation based on $\mathbf{S}$ and n is a user-defined value. CAINS uses $n = 6$ standard deviations for its anomaly detection algorithm. For measurements that pass the anomaly check, the Kalman filter uses the measurement to calculate the state estimate, $\hat{\mathbf{x}}(t_{i+1})$, and the updated covariance matrix, $\mathbf{P}$. If the measurement fails the anomaly check, the measurement is discarded and no measurement update occurs. Whether or not the measurement was anomalous, the algorithm saves the state values and uncertainty into the state structure and continues the loop.

3.4 State and Observation Models

The linearized $\mathbf{F}$ model used by the EKF is primarily made of the individual sensor's $\mathbf{F}$ matrices with few cross-sensor terms.

$$\mathbf{F} = \begin{bmatrix} \mathbf{F}_{\text{INS}} & \mathbf{F}_{\text{IMU}\times\text{INS}} & \mathbf{0}_{9\times2} & \mathbf{0}_{9\times2} \\ \mathbf{0}_{12\times9} & \mathbf{F}_{\text{IMU}} & \mathbf{0}_{12\times2} & \mathbf{0}_{12\times2} \\ \mathbf{0}_{2\times9} & \mathbf{0}_{2\times12} & \mathbf{F}_{\text{ST}} & \mathbf{0}_{2\times2} \\ \mathbf{0}_{2\times9} & \mathbf{0}_{2\times12} & \mathbf{0}_{2\times2} & \mathbf{F}_{\text{HT}} \end{bmatrix} \tag{42}$$

Before describing each submatrix, Table 2 defines Earth ellipsoid parameters necessary for the calculations.

$\mathbf{F}_{\text{INS}}$ is given by Equation (43), where the subscripts θ, v, ψ, and h represent the partial derivatives with respect to each state. Note that the order of states described by the state vector in Equation (40) has the position state elements broken up, such that the geodetic angles, ϕ and λ, are the first two states while the height h is last. The resulting $\mathbf{F}$ matrices describing the partial derivatives follow this state order. As

Table 2. Earth Parameters

Symbol	**Parameter (Units)**	**Value**
r	Ellipsoid semi-major radius (m)	6.378×10^6
e	Eccentricity	0.0818
Ω	Rotation rate (rad/s)	7.292×10^{-5}
R_E	Prime vertical radius of curvature (m)	$\frac{r}{\sqrt{1-e^2\sin^2(\phi)}}$
R_N	Meridian radius of curvature (m)	$\frac{r(1-e)}{\sqrt{1-e^2\sin^2(\phi)}^3}$
GM	Standard gravitational parameter (m^3/s^2)	3.986×10^{14}
J_2	Second degree zonal harmonic	1.083×10^{-3}

such, ϕ and λ are grouped together and represented by as a single unit, θ.

$$\mathbf{F}_{\text{INS}} = \begin{bmatrix} \mathbf{F}_{\theta\theta} & \mathbf{F}_{v\theta} & \mathbf{0}_{2\times3} & \mathbf{0}_{2\times1} \\ \mathbf{F}_{\theta v} & \mathbf{F}_{vv} & \mathbf{F}_{\psi v} & \mathbf{F}_{hv} \\ \mathbf{F}_{\theta\psi} & \mathbf{F}_{v\psi} & \mathbf{F}_{\psi\psi} & \mathbf{0}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{F}_{vh} & \mathbf{0}_{1\times3} & 0 \end{bmatrix} \tag{43}$$

The non-zero partial derivatives with respect to the θ states are

$$\mathbf{F}_{\theta\theta} = \begin{bmatrix} \frac{v_N \tan(\phi)}{\gamma_N} & \frac{-v_E \tan(\phi)}{\gamma_E} \\ 0 & 0 \end{bmatrix} \tag{44}$$

$$\mathbf{F}_{v\theta} = \begin{bmatrix} 0 & \frac{1}{\gamma_E} & 0 \\ \frac{-1}{\gamma_N} & 0 & 0 \end{bmatrix} \tag{45}$$

where γ_N and γ_E are shorthand representations of $R_N + h$ and $R_E + h$, respectively.

The partial derivatives with respect to the velocity states are

$$\mathbf{F}_{\theta v} = \begin{bmatrix} 2\Omega v_D \left(\sin(\phi) - \cos(\phi)\tan(\phi)\right) - [\mathbf{g}^n]_3 & 2\Omega v_E \cos(\phi) \\ -[\mathbf{g}^n]_3 & 2\Omega \left(v_D \sin(\phi - v_N \cos(\phi)\right) \\ -2\Omega v_N \left(\sin(\phi) - \cos(\phi)\tan(\phi)\right) - [\mathbf{g}^n]_3 & -2\Omega v_E \sin(\phi) \end{bmatrix} \tag{46}$$

$$\mathbf{F}_{vv} = \begin{bmatrix} \frac{v_D}{\gamma_N} & -2\left(\Omega \sin(\phi) + \frac{v_E \tan(\phi)}{\gamma_E}\right) & \frac{v_N}{\gamma_N} \\ 2\Omega \sin(\phi) + \frac{v_E \tan(\phi)}{\gamma_E} & \frac{v_D + v_N \tan(\phi)}{\gamma_E} & 2\Omega \cos(\phi) + \frac{v_E}{\gamma_E} \\ \frac{-2v_N}{\gamma_N} & -2\left(\Omega \cos(\phi) + \frac{v_E}{\gamma_E}\right) & 0 \end{bmatrix} \tag{47}$$

$$\mathbf{F}_{\psi v} = \left[\mathbf{C}_b^n \frac{\mathbf{\Delta v}}{\Delta t}\right]_\times \tag{48}$$

$$\mathbf{F}_{hv} = \begin{bmatrix} 0 & 0 & -2\left[\mathbf{g}^n\right]_3 \frac{1}{\gamma_N} \end{bmatrix}^{\mathrm{T}} \tag{49}$$

where the $[\cdots]_n$ operator represents the nth column vector of a matrix, the $[\cdots]_\times$ operator represents the skew-symmetric matrix of a vector, $\mathbf{C}_b^n$ is the DCM from the b-frame to n-frame, $\mathbf{\Delta v}$ is from the IMU measurement, and $\mathbf{g}^n$ is the gravitational mass attraction model rotated from the e-frame to the n-frame using the DCM $\mathbf{C}_e^n$,

$$\begin{aligned} \mathbf{g}^n &= \mathbf{C}_e^n \mathbf{g}^e \\ &= \mathbf{C}_e^n \left(-\mathbf{p}^e \frac{GM}{r^3} \; .\times \begin{bmatrix} 1 + 1.5J_2 \left(\frac{R_E}{r}\right)^2 \left(1 - 5\left(\frac{p_z^e}{r}\right)^2\right) + \Omega^2 p_x^e \\ 1 + 1.5J_2 \left(\frac{R_E}{r}\right)^2 \left(1 - 5\left(\frac{p_z^e}{r}\right)^2\right) + \Omega^2 p_y^e \\ 1 + 1.5J_2 \left(\frac{R_E}{r}\right)^2 \left(3 - 5\left(\frac{p_z^e}{r}\right)^2\right) \end{bmatrix} \right) \end{aligned} \tag{50}$$

where $\mathbf{p}^e$ is the e-frame position vector and the $.\times$ operator represents element-wise

matrix multiplication. The non-zero partial derivatives with respect to the tilt states are

$$\mathbf{F}_{\theta\psi} = \begin{bmatrix} 0 & \Omega\sin(\phi) \\ -\Omega\left(\sin(\phi) - 2\cos(\phi)\tan(\phi)\right) & 0 \\ 0 & \Omega\cos(\phi) \end{bmatrix} \tag{51}$$

$$\mathbf{F}_{v\psi} = \begin{bmatrix} 0 & \frac{1}{\gamma_E} & 0 \\ \frac{-1}{\gamma_N} & 0 & 0 \\ 0 & \frac{-\tan(\phi)}{\gamma_E} & 0 \end{bmatrix} \tag{52}$$

$$\mathbf{F}_{\psi\psi} = \begin{bmatrix} 0 & -\Omega\sin(\phi) - \frac{v_E\tan(\phi)}{\gamma_E} & \frac{v_N}{\gamma_N} \\ \Omega\sin(\phi) + \frac{v_E\tan(\phi)}{\gamma_E} & 0 & \Omega\cos(\phi) + \frac{v_E}{\gamma_E} \\ \frac{-v_N}{\gamma_N} & \Omega\cos(\phi) - \frac{v_E}{\gamma_E} & 0 \end{bmatrix} \tag{53}$$

Finally, the only non-zero partial derivative with respect to the altitude is the down velocity.

$$\mathbf{F}_{vh} = \begin{bmatrix} 0 & 0 & -1 \end{bmatrix} \tag{54}$$

The other sensor models, $\mathbf{F}_{IMU}$, $\mathbf{F}_{\mathrm{ST}}$, and $\mathbf{F}_{\mathrm{HT}}$, are based on the first order Gauss Markov biases,

$$\mathbf{F}_{\mathrm{IMU}} = \begin{bmatrix} \frac{-1}{\tau_{\mathrm{b}_a}}\mathbf{I}_{3\times3} & \frac{-1}{\tau_{\mathrm{b}_g}}\mathbf{I}_{3\times3} & \frac{-1}{\tau_{\mathrm{sf}_a}}\mathbf{I}_{3\times3} & \frac{-1}{\tau_{\mathrm{sf}_g}}\mathbf{I}_{3\times3} \end{bmatrix}_D \tag{55}$$

$$\mathbf{F}_{\mathrm{ST}} = \frac{-1}{\tau_{\mathrm{ST}}}\mathbf{I}_{2\times2} \tag{56}$$

$$\mathbf{F}_{\mathrm{HT}} = \frac{-1}{\tau_{\mathrm{HT}}}\mathbf{I}_{2\times2} \tag{57}$$

where the $[\cdots]_D$ operator indicates a diagonal matrix and the τ time constants are

given in Table 1. The only cross-sensor terms exist between the INS and the IMU.

$$\mathbf{F}_{\text{IMU}\times\text{INS}} = \begin{bmatrix} \mathbf{0}_{2\times3} & \mathbf{0}_{2\times3} & \mathbf{0}_{2\times3} & \mathbf{0}_{2\times3} \\ \mathbf{F}_{\text{b}_a v} & \mathbf{0}_{3\times3} & \mathbf{F}_{\text{sf}_a v} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & \mathbf{F}_{\text{b}_g \psi} & \mathbf{0}_{3\times3} & \mathbf{F}_{\text{sf}_g \psi} \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{0}_{2\times3} & \mathbf{0}_{2\times3} & \mathbf{0}_{2\times3} & \mathbf{0}_{2\times3} \\ \mathbf{C}_b^n & \mathbf{0}_{3\times3} & \mathbf{C}_b^n \left[\frac{\mathbf{\Delta v}}{\Delta t}\right]_D & \mathbf{0}_{3\times3} \\ \mathbf{0}_{3\times3} & -\mathbf{C}_b^n & \mathbf{0}_{3\times3} & -\mathbf{C}_b^n \left[\frac{\mathbf{\Delta v}}{\Delta t}\right]_D \\ \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} & \mathbf{0}_{1\times3} \end{bmatrix} \tag{58}$$

There are two linearized $\mathbf{H}$ models used by the EKF, one for the star tracker and the other for the HAV tracker. While both are similar sensors, they use different measurements, which lead to different models for both. The star tracker's measurement is the image coordinates in the p-frame, $\mathbf{y}^p$,

$$\mathbf{h}_{\text{ST}} = \mathbf{y}^p = \begin{bmatrix} y_x^p \\ y_y^p \end{bmatrix} \tag{59}$$

The linearized model for the star tracker is then

$$\mathbf{H}_{\text{ST}} = \begin{bmatrix} \mathbf{H}_{\theta\text{ST}} & \mathbf{0}_{2\times7} & \mathbf{0}_{2\times12} & \mathbf{H}_{\text{b}_{\text{ST}}\text{ST}} & \mathbf{0}_{2\times2} \end{bmatrix} \tag{60}$$

where $\mathbf{H}_{\theta\text{ST}}$ and $\mathbf{H}_{\text{b}_{\text{ST}}\text{ST}}$ are

$$\mathbf{H}_{\theta\text{ST}} = \begin{bmatrix} \frac{u_y^n}{u_z^c}\left(\tan(\phi) - y_x^p\right) & \frac{y_x^p u_x^n + u_z^n}{u_z^c} \\ \frac{-u_x^n \tan(\phi) - y_y^p u_y^n - u_z^n}{u_z^c} & \frac{y_y^p u_x^n}{u_z^c} \end{bmatrix} \tag{61}$$

$$\mathbf{H}_{\mathrm{b_{ST}ST}} = \begin{bmatrix} 1 + y_x^{p2} & -y_y^p \sin(\mathrm{b}_a) + y_x^p y_y^p \cos(\mathrm{b}_a) \\ 1 + y_x^p y_y^p & y_x^p \sin(\mathrm{b}_a) + \cos(\mathrm{b}_a) + y_y^{p2} \cos(\mathrm{b}_a) \end{bmatrix} \tag{62}$$

where $\mathbf{u}^n$ and $\mathbf{u}^c$ are the star tracker's unit pointing vectors in the n-frame and c-frame, respectively, and b_a is the star tracker's azimuth bias.

The HAV tracker outputs azimuth and elevation angles in the c-frame defined by the position vector from the RPA to HAV, $\mathbf{p}^c$,

$$\mathbf{h}_{\mathrm{HT}} = \begin{bmatrix} az \\ el \end{bmatrix} = \begin{bmatrix} \arctan\left(\dfrac{p_x^c}{p_z^c}\right) \\ \arcsin\left(\dfrac{p_y^c}{|\mathbf{p}^c|}\right) \end{bmatrix}. \tag{63}$$

The linearized model for HAV tracker is

$$\mathbf{H}_{\mathrm{HT}} = \begin{bmatrix} \mathbf{H}_{\theta\mathrm{HT}} & \mathbf{0}_{2\times3} & \mathbf{H}_{\psi\mathrm{HT}} & \mathbf{H}_{h\mathrm{HT}} & \mathbf{0}_{2\times12} & \mathbf{0}_{2\times2} & \mathbf{H}_{\mathrm{b_{HT}HT}} \end{bmatrix} \tag{64}$$

where $\mathbf{H}_{\theta\mathrm{HT}}$, $\mathbf{H}_{\psi\mathrm{HT}}$, $\mathbf{H}_{h\mathrm{HT}}$, and $\mathbf{H}_{\mathrm{b_{HT}HT}}$ are

$$\mathbf{H}_{\theta\mathrm{HT}} = \mathbf{A}(\mathbf{C}_n^c)_{\mathrm{bias}}(\mathbf{B} + \mathbf{C}_e^n \mathbf{D}) \tag{65}$$

$$\mathbf{H}_{\psi\mathrm{HT}} = \mathbf{A}(\mathbf{C}_n^c)_{\mathrm{bias}}[\mathbf{p}^n]_\times \tag{66}$$

$$\mathbf{H}_{h\mathrm{HT}} = -\mathbf{A}(\mathbf{C}_e^c)_{\mathrm{bias}}[\mathbf{C}_e^n]_{\mathbf{z}}^{\mathbf{T}}\,[\tag{67}$$

$$\mathbf{H}_{\mathrm{b_{HT}HT}} = \begin{bmatrix} 1 & \dfrac{p_x^c p_y^c}{p_x^{c\,2} + p_y^{c\,2}}(\cos(\mathrm{b}_a) - \sin(\mathrm{b}_a)) \\ 0 & \dfrac{p_x^c \sin(\mathrm{b}_a) + p_z^c \cos(\mathrm{b}_a)}{\sqrt{p_x^{c\,2} + p_z^{c\,2}}} \end{bmatrix} \tag{68}$$

where $\mathbf{p}^n$ is the RPY to HAV position vector in the n-frame, b_a is the HAV tracker's azimuth bias, $\mathbf{C}_n^c$ is the n-frame to c-frame DCM, $\mathbf{C}_e^c$ is the e-frame to c-frame

DCM, and $(\mathbf{C}_n^c)_{\text{bias}}$ and $(\mathbf{C}_e^c)_{\text{bias}}$ are the c-frame DCMs with the bias-correction. The matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{D}$ are defined as

$$\mathbf{A} = \begin{bmatrix} \frac{p_z^c}{p_x^{c\,2} + p_z^{c\,2}} & 0 & \frac{-p_x^c}{p_x^{c\,2} + p_z^{c\,2}} \\ \frac{-p_x^c p_y^c}{|\mathbf{p}|^2 \sqrt{p_x^{c\,2} + p_z^{c\,2}}} & \frac{p_x^{c\,2} + p_z^{c\,2}}{|\mathbf{p}|^2 \sqrt{p_x^{c\,2} + p_z^{c\,2}}} & \frac{-p_y^c p_z^c}{|\mathbf{p}|^2 \sqrt{p_x^{c\,2} + p_z^{c\,2}}} \end{bmatrix} \tag{69}$$

$$\mathbf{B} = \begin{bmatrix} p_y^n \tan(\phi) & p_z^n \\ -p_x^n \tan(\phi) - p_z^n & 0 \\ p_y^n & -p_x^n \end{bmatrix} \tag{70}$$

$$\mathbf{D} = \begin{bmatrix} (R_E + h)[\mathbf{C}_e^n]_{2,1} & (R_E + h)[\mathbf{C}_e^n]_{1,1} + \frac{R_E e^2 \sin(\phi)\cos(\phi)}{1 - e^2 \sin^2(\phi)}[\mathbf{C}_e^n]_{3,1} \\ (R_E + h)[\mathbf{C}_e^n]_{2,2} & (R_E + h)[\mathbf{C}_e^n]_{1,2} + \frac{R_E e^2 \sin(\phi)\cos(\phi)}{1 - e^2 \sin^2(\phi)}[\mathbf{C}_e^n]_{3,2} \\ (R_E(1 - e^2) + h)[\mathbf{C}_e^n]_{2,3} & (R_E(1 - e^2) + h)[\mathbf{C}_e^n]_{1,3} + \frac{R_E e^2 \sin(\phi)\cos(\phi)}{1 - e^2 \sin^2(\phi)}[\mathbf{C}_e^n]_{3,3} \end{bmatrix} \tag{71}$$

where $[\cdots]_{i,j}$ is the i, j element of a matrix.

3.5 Generating New Data for CAINS

For this research, new star tracker and HAV tracker measurement data had to be generated. For the star tracker data, the star catalog that was provided by the sponsor was used, which included 12443 stars. For the HAV data, an HAVs trajectory was generated similar to the RPAs true flight path but at an altitude of about 20 km. With the star table and the HAV trajectory, 20 measurement data files were generated using the RPAs FOV and differing measurement frequencies. Besides the changing measurement frequency of the CNS sensors, all other parameters were fixed. Table 3 captures the 20 different measurement frequencies tested. These intervals were selected to capture a wide variety and to determine their effects on the navigation solution.

Table 3. Sensor measurement frequencies between simulations. Only the CNS sensors had varying measurement intervals while the IMU and altimeter stayed constant.

Sensor	Measurement Interval Δt (s)
IMU	0.01
Altimeter	30
CNS	1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 20, 30, 40, 50

Additional simulations were tested with CNS measurements beyond 50 s intervals. However the true attitude of the RPA fluctuates throughout the simulation so the fixed CNS sensors miss the HAV measurements about 60% of the time. When the CNS measurement frequency was set over 50 s apart, the RPA missed practically all the HAV measurements so the navigation estimates drifted. As such, those results were excluded from this paper.

This effort only required new star and HAV measurements. The IMU and altimeter measurements were left untouched as they were provided by the sponsor. The IMU provided measurements every $\Delta t = 0.01$ s and the altimeter every $\Delta t = 30$ s.

3.6 Chapter III Summary

This chapter provided a description of the scenarios simulated and the tool used to simulate the scenarios. The general simulation of the RPA making stellar and HAV observations was described in Section 3.1 as well as the specific scenarios to be compared in the results. The process flow for CAINS was explained in Section 3.3, which was used to run the simulations. The EKF models for the sensors were shown in Section 3.4. Finally, new sensor data generation for the scenarios was discussed in Section 3.5. The next chapter will show the simulation results and analyze the differences between scenarios.

IV. Results

This chapter discusses the results of the CAINS tool in estimating the navigation accuracies in the three different scenarios: star tracking only, HAV tracking only, and then star and HAV tracking. Each scenario will be discussed individually first, with the results of Scenario 1 in Section 4.1, Scenario 2 in Section 4.2, and Scenario 3 in Section 4.3. The 20 different simulations per scenario are analyzed to determine the effect of increasing the time between observations on navigation accuracy. Finally Section 4.4 compares the results between scenarios to determine the effect of the different sensor configurations on navigation accuracy.

4.1 Scenario 1: Star Tracking Only

In Scenario 1, only the star tracker and altimeter were used to aid the INS[1]. Figures 8 through 13 show the navigation state position, velocity, and attitude estimates of one sample run. The sample taken is the first simulation run in this scenario where the star tracker measurements are observed every $\Delta t = 1$ s. From Figure 8 which shows the the truth vs estimated LLH position, it can be seen that the latitude and longitude drifts over the 10 minute simulation while the altitude stays within a reasonable level of accuracy. This is expected since position updates are unavailable with star tracking, so the estimate will drift with the IMU, while the altimeter provides updates for the altitude. Even with the drift, the errors stay within the 3σ bounds as shown in Figure 9.

Similarly, Figure 10 shows the truth vs estimated NED velocities and Figure 11 shows the velocity estimate errors. The results are interesting because the velocities

[1]For the simulation results, the INS's initial conditions for the position, velocity, and attitude uncertainty was set fairly high, using 1σ values of 200 m, 20 m/s, and 1 mrad, respectively. This resulted in heavy drift errors even for a navigation grade IMU.

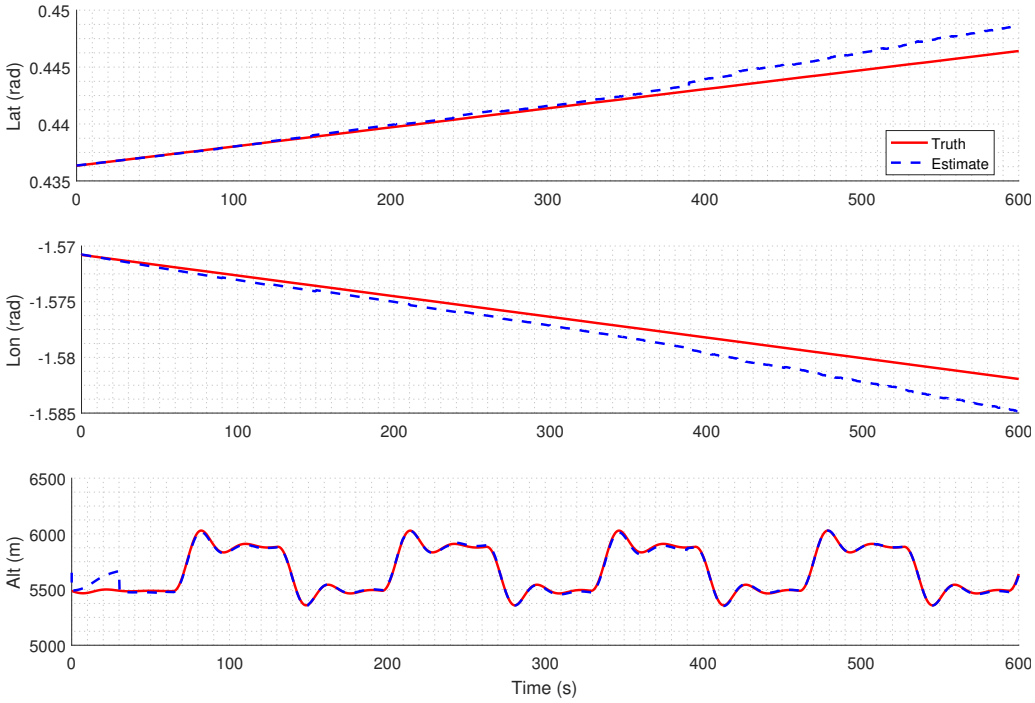


Figure 8. Sample plot from Scenario 1, truth vs estimate of the LLH position of the first simulation (dt = 1).

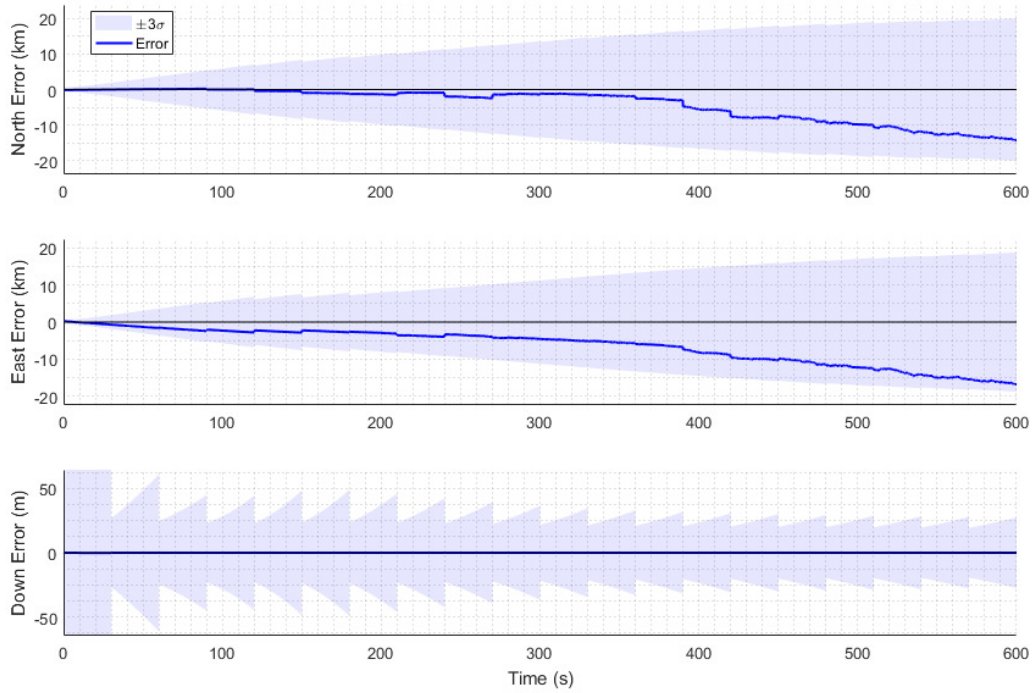


Figure 9. Sample plot from Scenario 1, position estimate errors of the first simulation (dt = 1) with 3σ uncertainty.

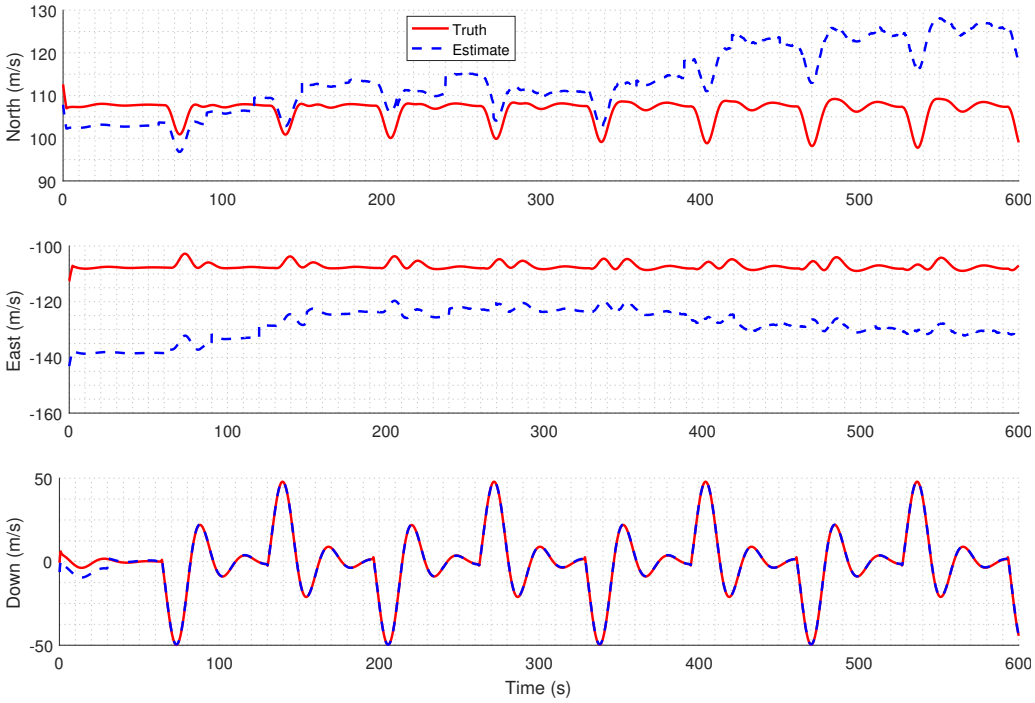


Figure 10. Sample plot from Scenario 1, truth vs estimate of the NED velocity of the first simulation (dt = 1).

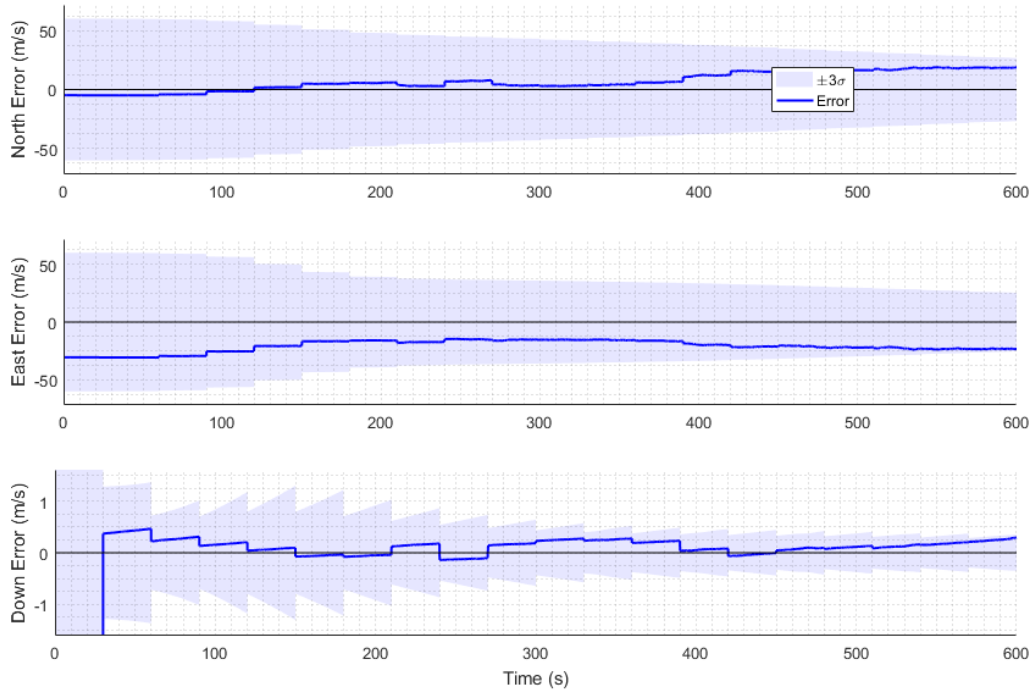


Figure 11. Sample plot from Scenario 1, velocity estimate errors of the first simulation (dt = 1) with 3σ uncertainty.

do not drift without bound as expected. The 3σ uncertainty for the North and East velocities taper inward over the simulation. This might be due to the altimeter keeping track of the down position.

Lastly, Figure 12 shows the truth vs estimated RPY attitude and Figure 13 shows the attitude estimate errors. These attitude errors described are the errors from the true Euler angles of the RPA. As expected, the attitude is tracked quite accurately since the purpose of the star tracker is to provide attitude updates. While the attitude error uncertainties still drift, the scale is small even at the end of the 10 minute simulation.

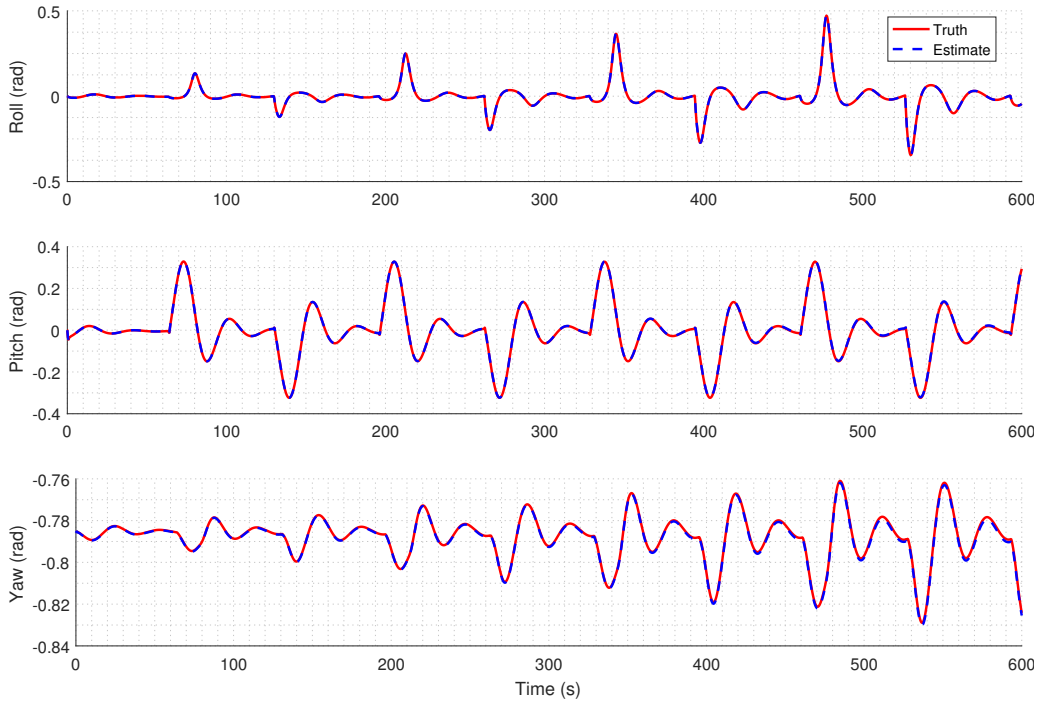


Figure 12. Sample plot from Scenario 1, truth vs estimate of the RPY attitude of the first simulation (dt = 1).

The sample plots presented show a single case of the twenty simulations run in this scenario. To effectively compare the twenty simulations, the results are summarized in more efficient measures. One such statistical measure is the root sum square (RSS) as shown in Equation (72), where x_i, y_i, and z_i are the ith navigation state estimate

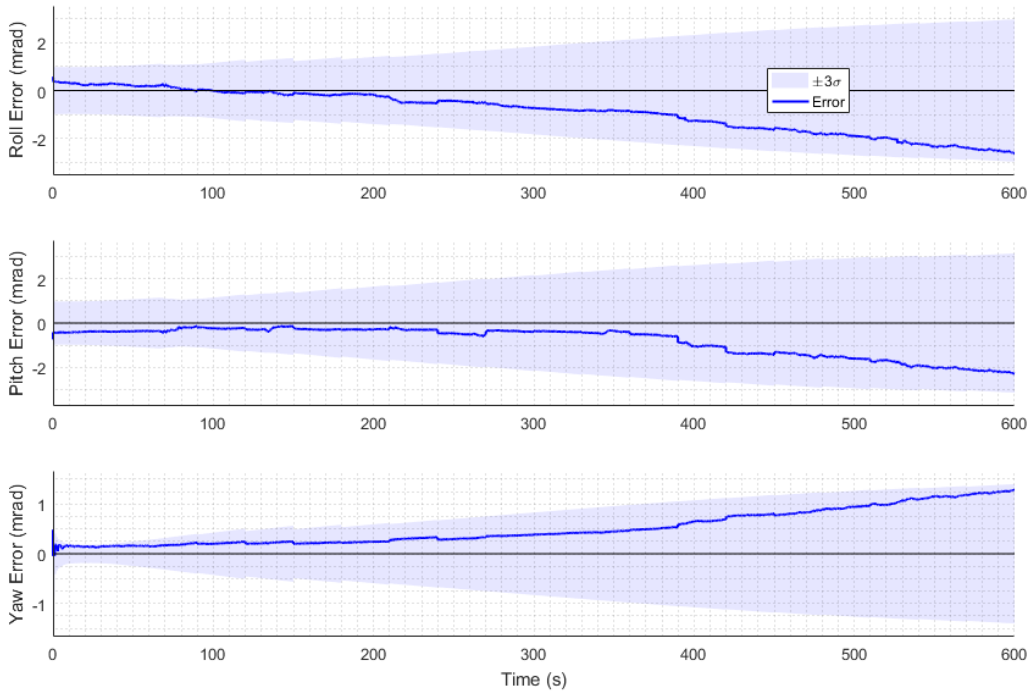


Figure 13. Sample plot from Scenario 1, attitude estimate errors of the first simulation (dt = 1) with 3σ uncertainty.

errors.

$$RSS = \sqrt{x_i^2 + y_i^2 + z_i^2} \tag{72}$$

The RSS of the errors combine the navigation state errors for position, velocity, or attitude in each of their three axis into a single value that describes the absolute distance of the error from zero [15]. Figure 14 shows a sample plot of the RSS of the position error from the first simulation run. The first 90 s are ignored as the EKF stabilizes to its steady state. The position error drift is also apparent in the RSS error.

It would still be inconvenient to show every RSS plot per simulation because it's time dependent. This time dependency can be removed by generating a cumulative distribution function (CDF) from the RSSs by determining how many measurements fall below a specific error value verses the total number of measurements in that run.

The CDF shows the probability of the state estimate error being equal to or less than a given value. Figure 15 shows a sample CDF of the RSS position error from Figure 14.

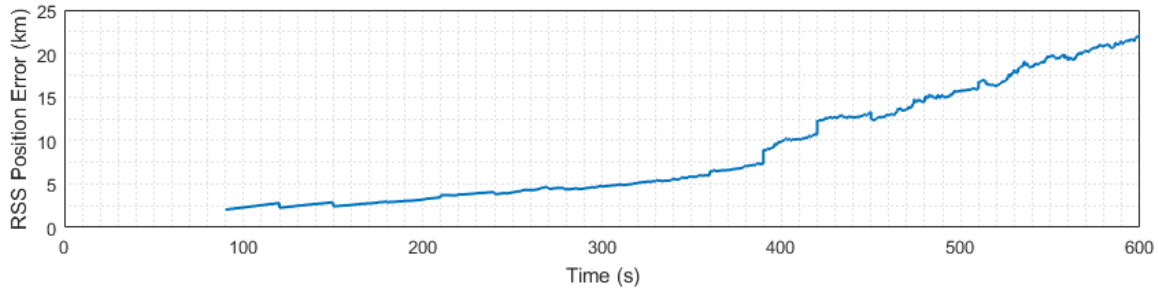


Figure 14. Sample RSS of the position estimate errors for Scenario 1 from the first simulation (dt = 1). The RSS is calculated after 90 s to allow the EKF to initialize.

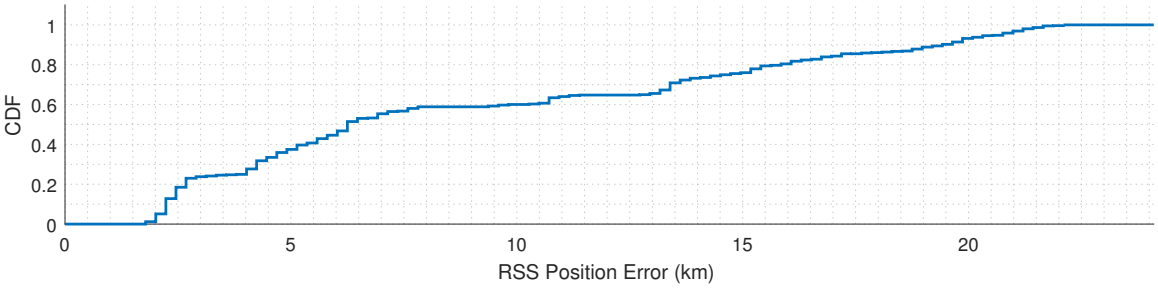


Figure 15. Sample CDF from Scenario 1 of the first simulation (dt = 1). The CDF was generated based on the RSS of the position estimate errors from Figure 14.

Finally by transforming the CDF into a single error bar, multiple simulations can be displayed side-by-side for a proper comparison and analysis of the data between runs. Figures 16 through 18 show exactly that, by generating a CDF per simulation for the position, velocity, and attitude RSS errors and transforming the CDFs into error bars. The error bars encapsulate an 80% confidence interval, from 10% probability at the bottom of the bar to 90% at the top. It also shows the 50% median point, which is useful to show how much of that specific run's results are above or below the median.

Now the filter estimates among different runs can be easily compared. Based on the CDF error bars in Figures 16 through 18, there doesn't appear to be a strong re-

lationship between the navigation accuracy and measurement observation frequency. The position, velocity, and attitude estimate errors vary widely among all the runs. There is a slight increase in the size of the error bars in the position and attitude errors for some of the runs as the measurement Δt's increase, but it's not consistent.

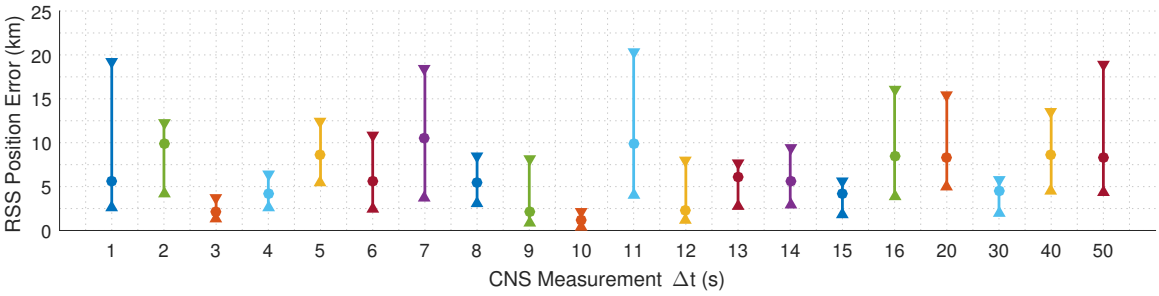


Figure 16. CDF error bars of the RSS position errors from Scenario 1.

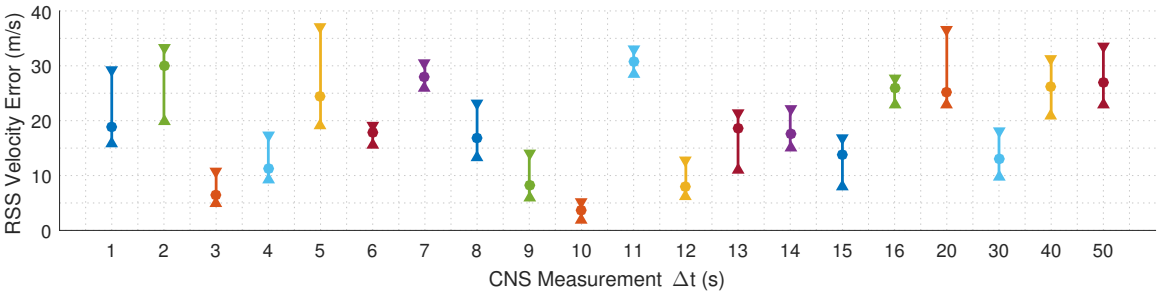


Figure 17. CDF error bars of the RSS velocity errors from Scenario 1.

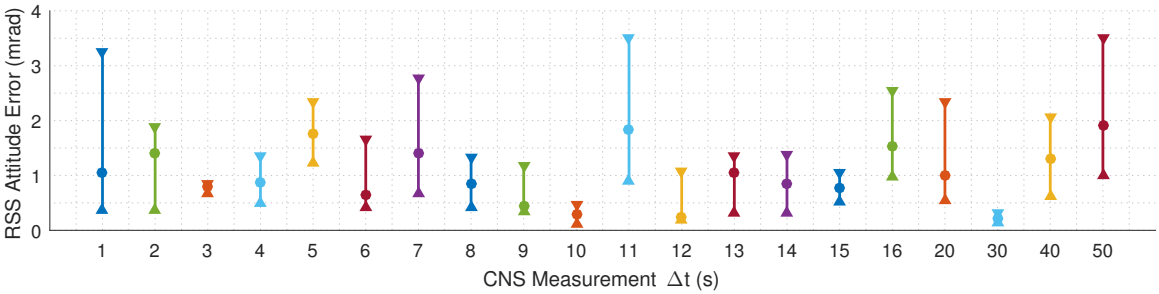


Figure 18. CDF error bars of the RSS attitude errors from Scenario 1.

Another useful statistical measure of accuracy, specifically for position, is 3-D root

mean square (3DRMS) as given by Equation (73),

$$3DRMS = \sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2 + y_i^2 + z_i^2} \tag{73}$$

where x_i, y_i, and z_i are the ith navigation state estimate error and n is the total number of measurements throughout the run. The 3DRMS value describes the overall 3-D position accuracy, combining the mean and standard deviations of the estimate errors from all three directional axes [11]. What's most useful about 3DRMS is that it summarizes the positional accuracy of an entire simulation run into a single point. It's similar to the error bars but even more compact by showing one point rather than a confidence interval.

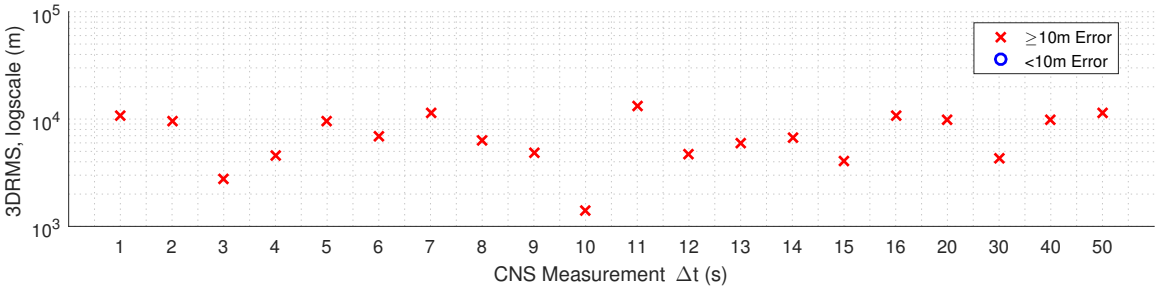


Figure 19. 3DRMS position accuracy for Scenario 1. Note the log-scale y-axis.

Figure 19 shows the 3DRMS values of the position accuracy of all the simulations in Scenario 1. Again, there is seemingly no correlation between measurement observation frequencies and position accuracy. The figure's legend depicts different markers for "good" simulations marked by ○'s versus "bad" simulations marked by ×'s. This threshold was arbitrarily set at 10 m of 3DRMS accuracy. In Scenario 1, due to the position estimate drift, all the 3DRMS values are over this threshold. The rest of this paper will only be discussed in terms of the CDF error bars and 3DRMS position estimates.

The last thing to note are the anomalous measurements flagged by the EKF.

Figure 20 shows the percentage of anomalous measurements per run flagged by the residual monitoring anomaly detector for each sensor. Similar to the good vs bad markers made for the 3DRMS plot, an arbitrary threshold was set at 20%. In this scenario, none of the sensor measurements were anomalous so this figure is not worth much discussion, but it will be important for the other two scenarios.

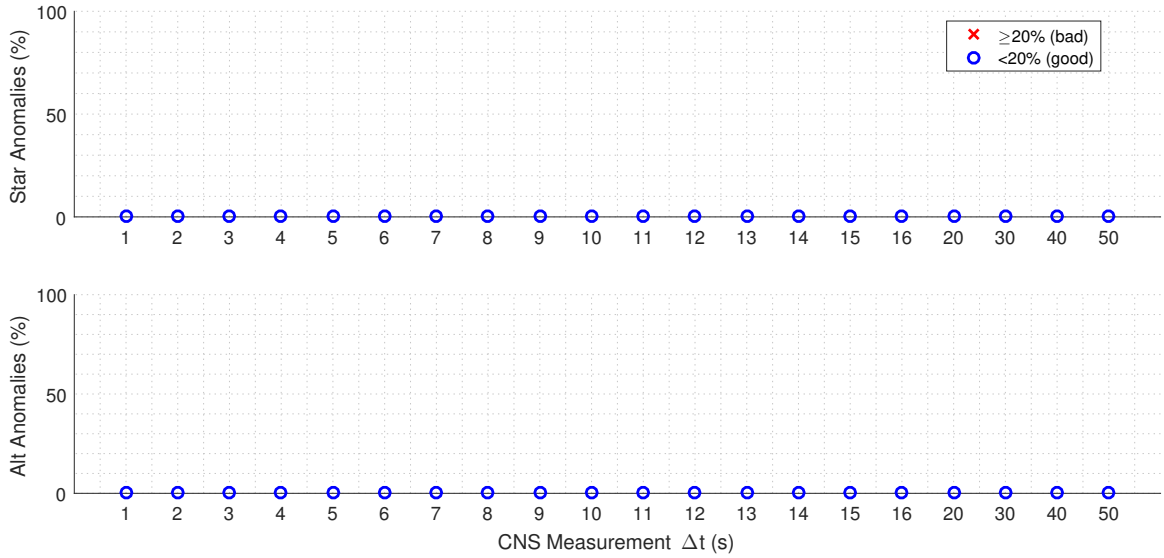


Figure 20. Measurement anomalies for each sensor in Scenario 1. No measurements were flagged as anomalous.

4.2 Scenario 2: HAV Tracking Only

In Scenario 2, only the HAV tracker and altimeter were used to aid the INS. Figures 21 through 23 show the sample plots of the first simulation run of this scenario. The truth vs estimate navigation states weren't shown as in Scenario 1 due to the small scales of the error, so those figures weren't worthwhile. These sample plots are only shown for reference. Most of the analysis will be spent on the RSS and 3DRMS plots.

Before showing the results, it's necessary to review the anomalies that the algorithm detected first. Figure 24 shows that the HAV tracker's measurements were

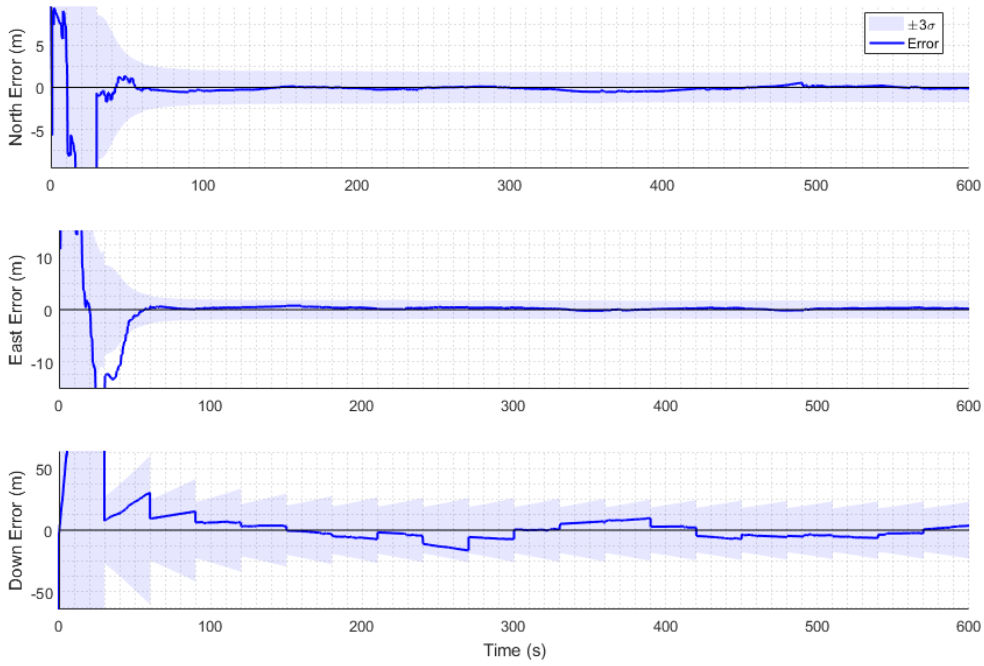


Figure 21. Sample plot from Scenario 2, position estimate errors of the first simulation (dt = 1) with 3σ uncertainty.

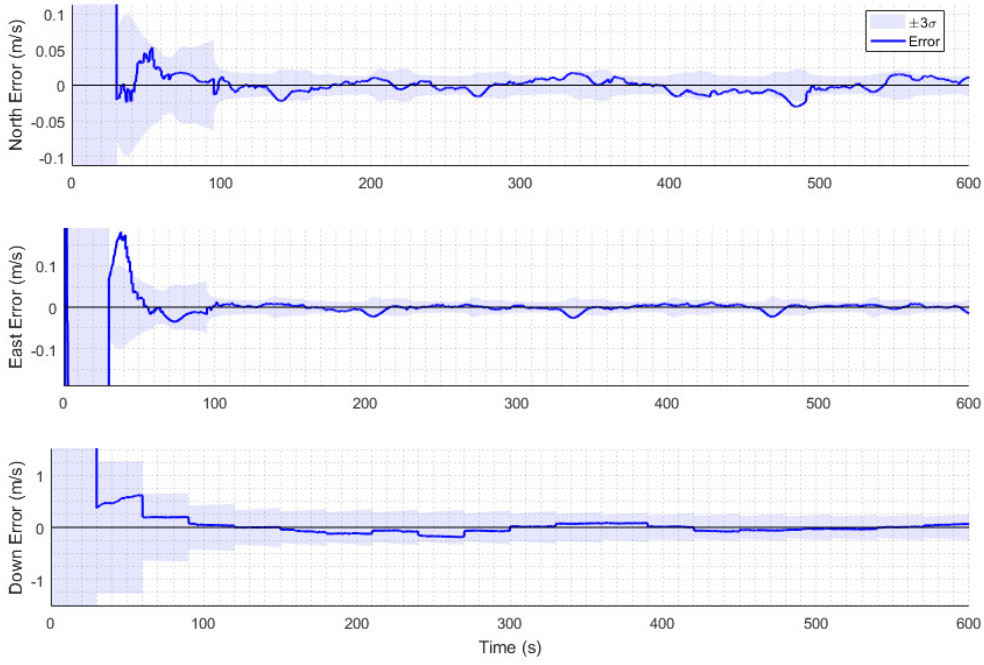


Figure 22. Sample plot from Scenario 2, velocity estimate errors of the first simulation (dt = 1) with 3σ uncertainty.

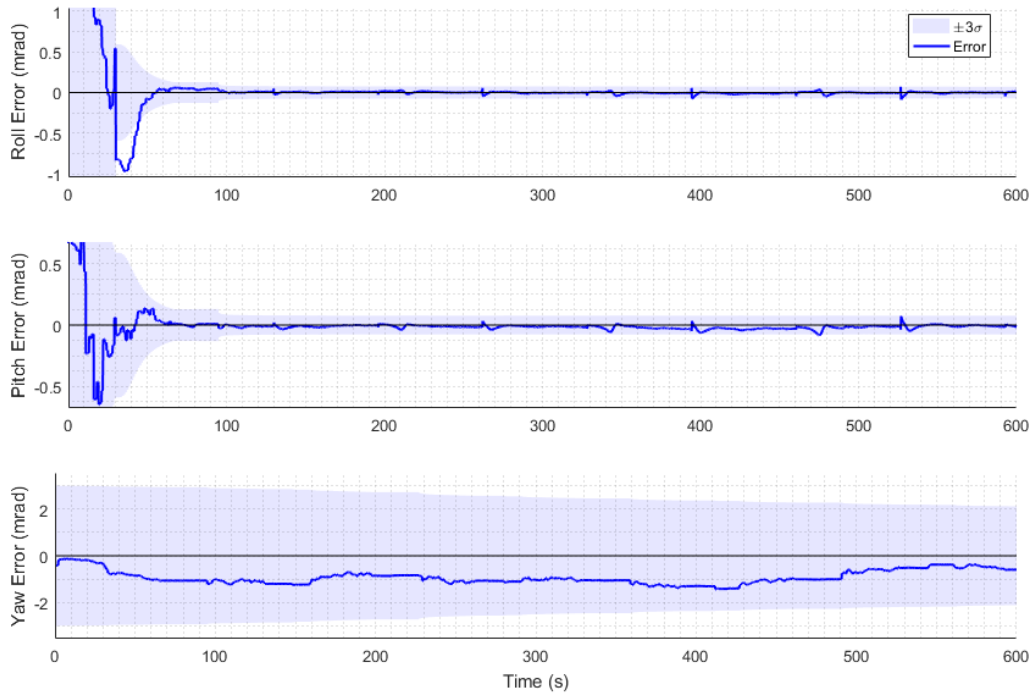


Figure 23. Sample plot from Scenario 2, attitude estimate errors of the first simulation (dt = 1) with 3σ uncertainty.

bad for simulation runs 10, 17 and 20 (corresponding with measurement Δt's of 10, 20 and 50 s) and the altimeter's measurements were bad for simulation runs 10 and 14. It's uncertain why the anomaly detection algorithm does this. As described in Section 3.5, each HAV measurement file is generated from the same HAV flight trajectory. The only difference between the measurement files is the different observation times. The anomalies with the altimeter is even more confusing since the exact same measurement file is read for all the simulation runs.

Figure 25 shows the 3DRMS accuracies of all the simulation runs in Scenario 2. The top subplot captures all the 3DRMS accuracies, including the bad ones. There is a direct correlation between the anomalous runs found in Figure 24 and the runs with large errors. This is because so many of the measurements in those runs are discarded so the IMU drifts with few updates.

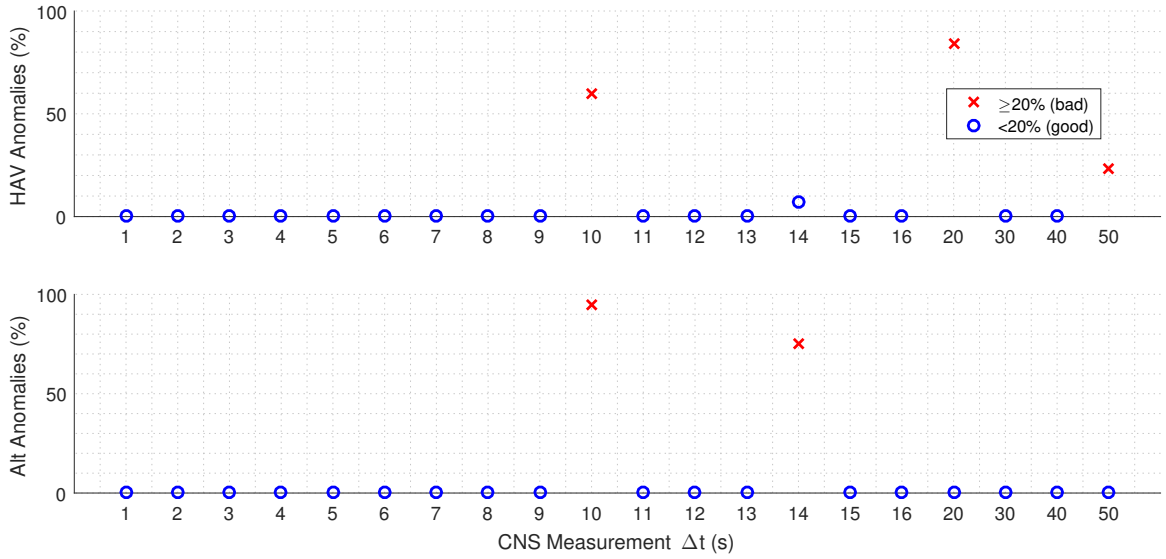


Figure 24. Measurement anomalies for each sensor in Scenario 2. The HAV tracker had bad measurements for simulations 10, 17, and 20 and the altimeter had bad measurements for simulations 10 and 14.

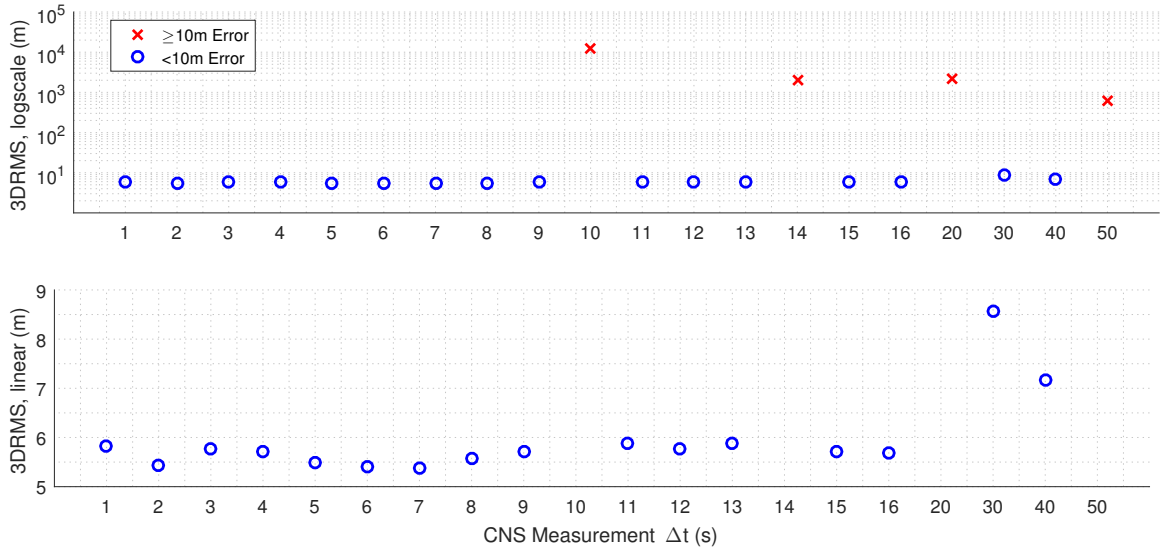


Figure 25. 3DRMS for Scenario 2. Both subfigures show the same data, but the top subfigure includes all simulation results whereas the bottom subfigure is scaled to just the "good" simulations with a 3DRMS under 10 m.

The bottom subplot of Figure 25 shows a closer view of only the good simulations. Except for the last two points, there is basically no relationship between 3DRMS accuracies and measurement observation times as with Scenario 1. The last two simulations are the only indication that the accuracies fall, but instead of trending

up to to those levels of accuracy, they suddenly jump from a steady average. These results may be outliers.

Figures 26 through 28 show the CDF error bars of the RSS position, velocity, and attitude estimate errors, respectively. The simulations with bad 3DRMS values are ignored. The position and velocity plots don't show any effect of increasing measurement observation times, but the attitude error RSS does show a slight drop in attitude accuracy towards the later simulation runs. The last two simulations with outlying 3DRMS results are also apparent in the position and velocity RSS plots, yet counterintuitively those same two runs have very accurate attitude error bars.

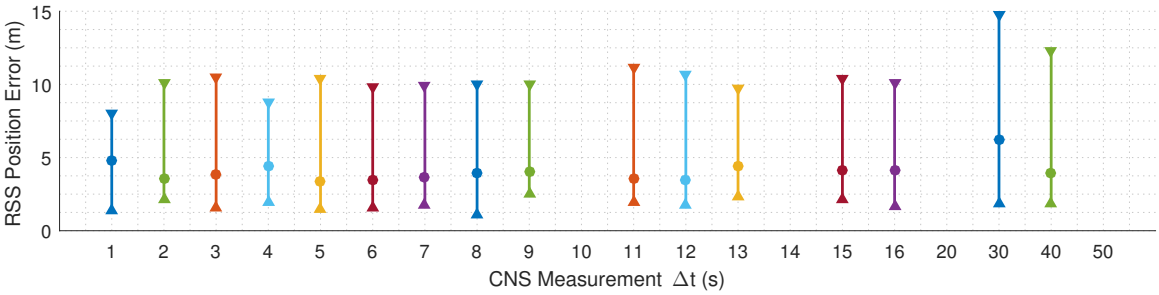


Figure 26. CDF error bars of the RSS position errors from Scenario 2. Note that only the simulations with a 3DRMS below 10 m are shown.

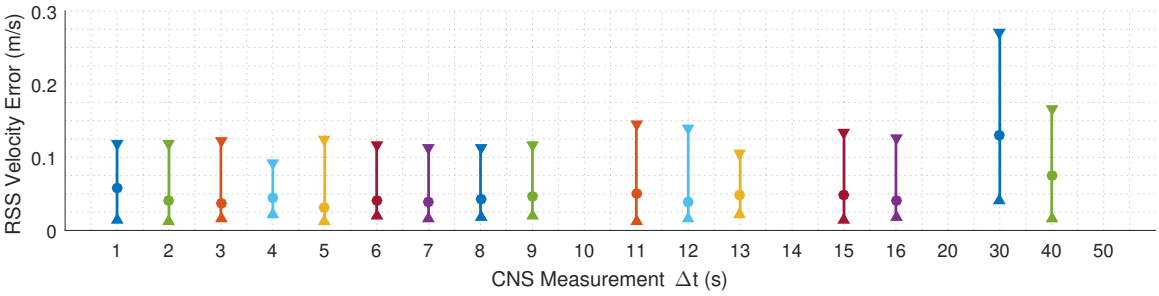


Figure 27. CDF error bars of the RSS velocity errors from Scenario 2. Note that only the simulations with a 3DRMS below 10 m are shown.

4.3 Scenario 3: Combined Star and HAV Tracking

In Scenario 3, both the star tracker and HAV tracker are used with the altimeter for aiding the INS. As in the previous scenarios, the sample plots of the error estimates

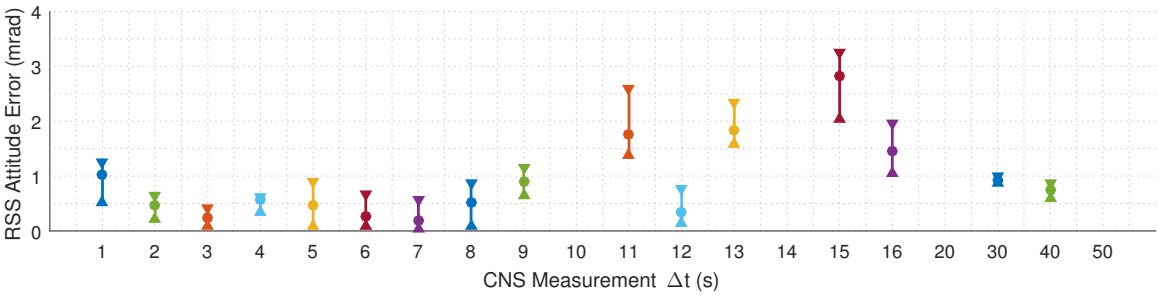


Figure 28. CDF error bars of the RSS attitude errors from Scenario 2. Note that only the simulations with a 3DRMS below 10 m are shown.

are shown in Figures 29 through 31 and are shown for reference.

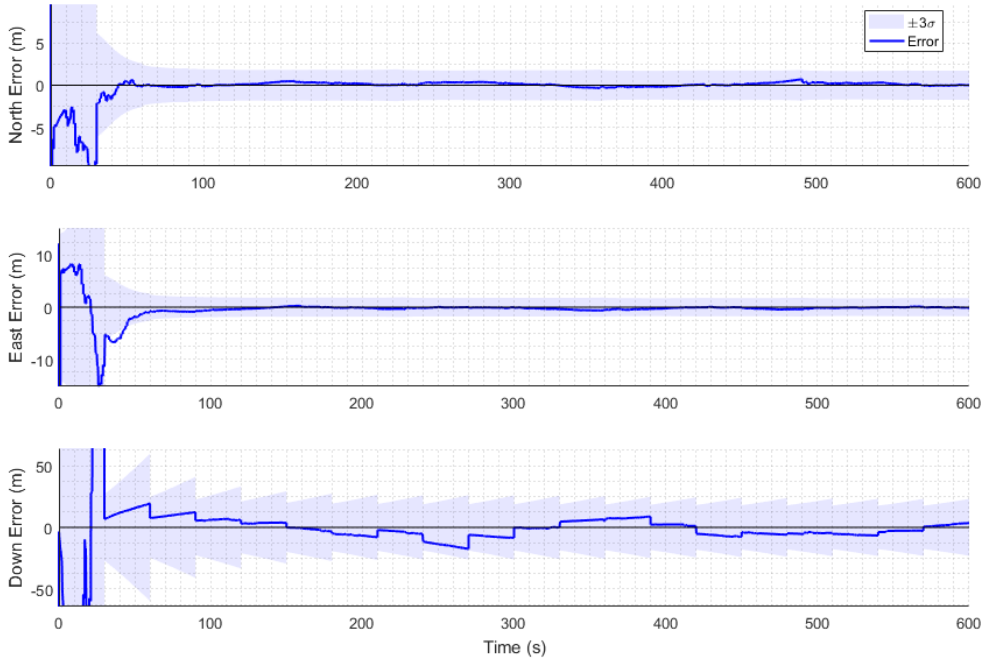


Figure 29. Sample plot from Scenario 3, position estimate errors of the first simulation (dt = 1) with 3σ uncertainty.

As with Scenario 2, the anomalies will be looked at first. Figure 32 shows which runs had bad data per sensor. In this scenario, none of the star tracker measurements were bad. There were 4 bad simulations for the HAV tracker: simulations 15, 16, 17, and 19, corresponding to observation measurement Δt's of 15, 16, 20, and 40. Of those, simulation 17 was also a bad simulation from Scenario 2. There were 2 bad

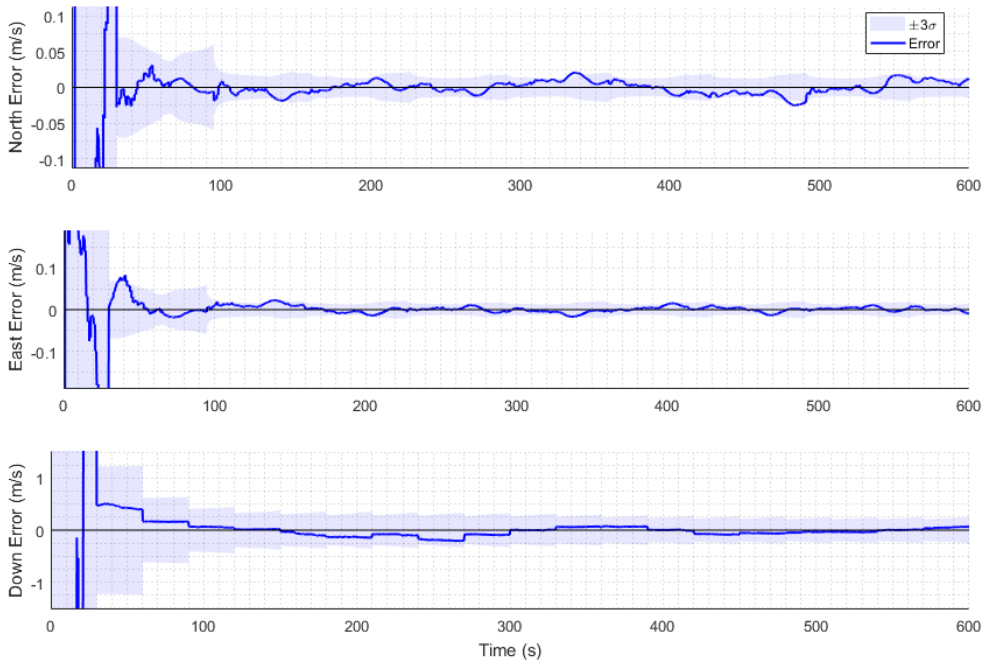


Figure 30. Sample plot from Scenario 3, velocity estimate errors of the first simulation (dt = 1) with 3σ uncertainty.

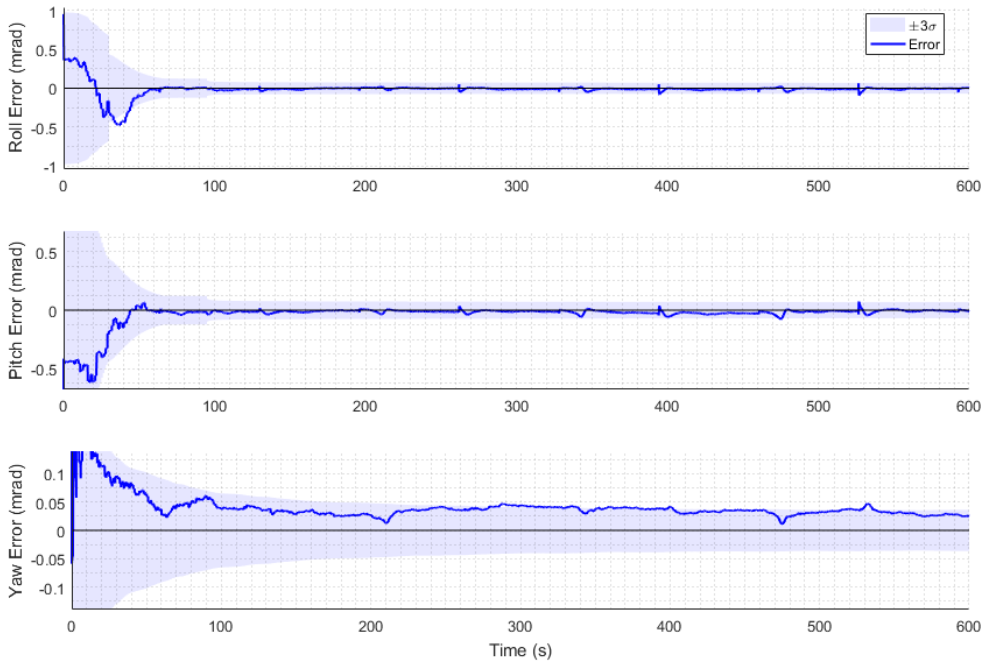


Figure 31. Sample plot from Scenario 3, attitude estimate errors of the first simulation (dt = 1) with 3σ uncertainty.

simulations for the altimeter: simulations 10 and 13. Of those, simulation 10 was also a bad simulation from Scenario 2. There may be a correlation between measurement time Δt's and anomalies found, as most of the bad runs are around the middle runs.

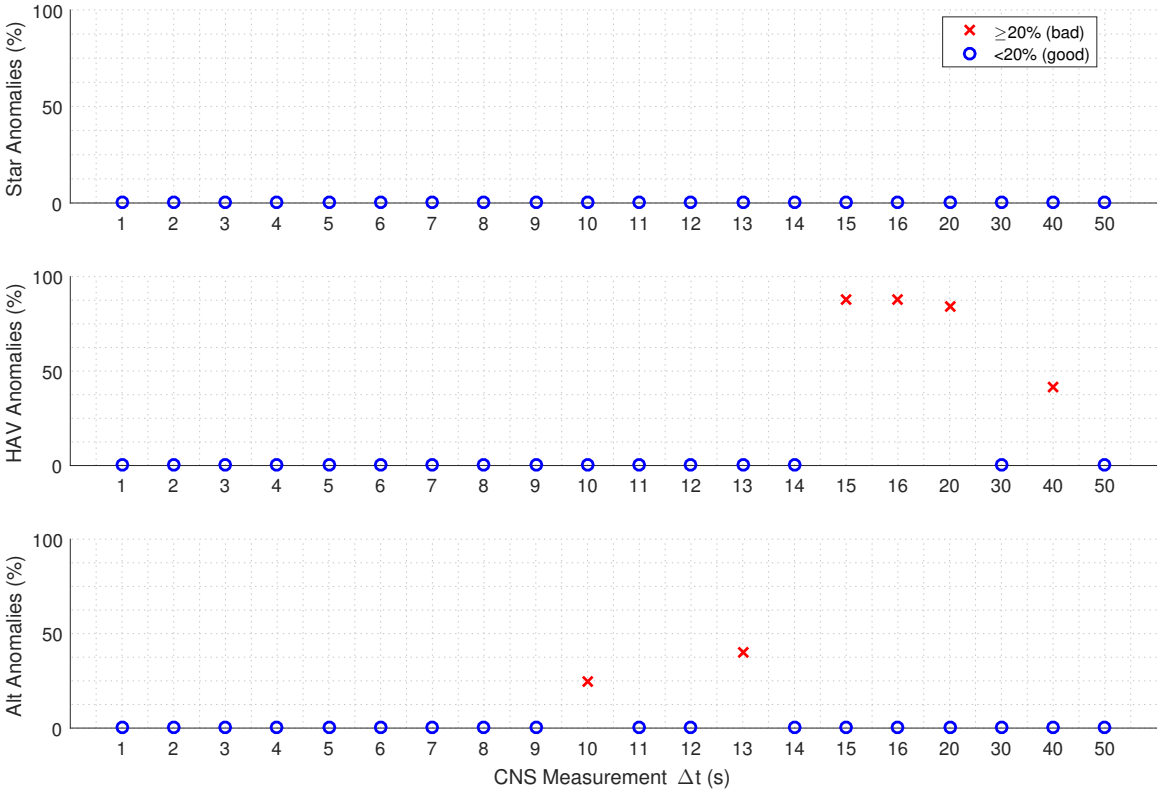


Figure 32. Measurement anomalies for each sensor in Scenario 3. The HAV tracker had bad measurements for simulations 15, 16, 17, and 19. The altimeter had bad measurements for simulation 10 and 13.

Figure 33 plots all the 3DRMS accuracies of this scenario. As with Scenario 2, the navigation estimates associated with the anomalous simulations experience large drift errors. These simulations are ignored in later results. Taking a closer look at the good simulations also shows no trend in accuracy due to measurement frequency.

Figures 34 through 36 show the RSS error bars of the position, velocity, and attitude estimates, respectively. There is no visible relationship between measurement observation Δt's and estimate accuracies except the last two attitude error results

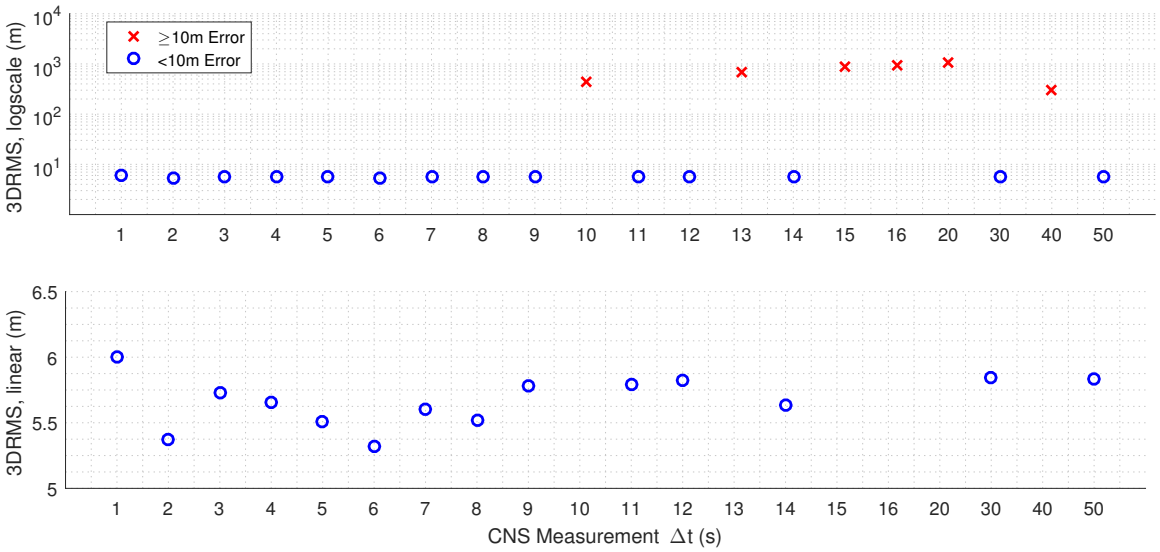


Figure 33. 3DRMS for Scenario 3. Both subfigures show the same data, but the top subfigure includes all simulation results whereas the bottom subfigure is scaled to just the "good" simulations with a 3DRMS under 10 m.

which seem to show slightly worse attitude estimates.

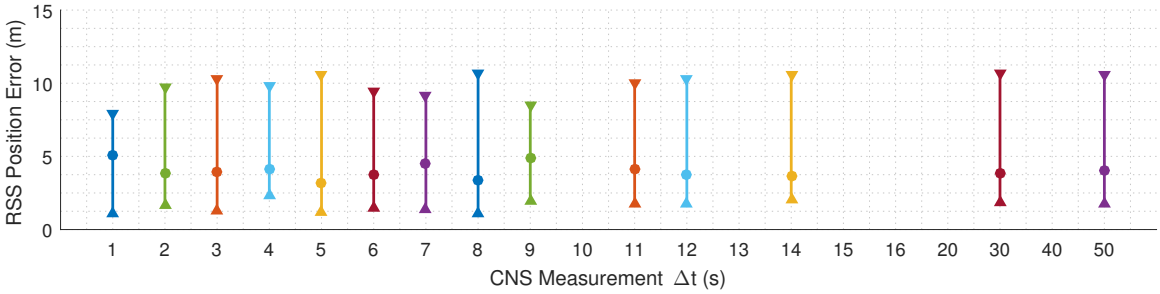


Figure 34. CDF error bars of the RSS position errors from Scenario 3. Note that only the simulations with a 3DRMS below 10 m are shown.

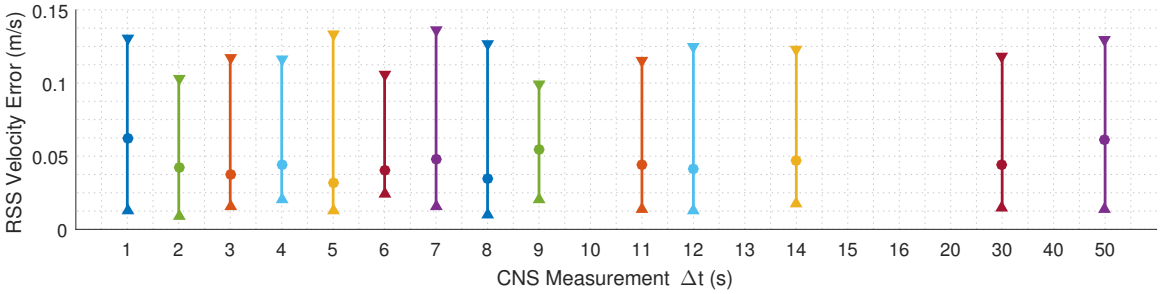


Figure 35. CDF error bars of the RSS velocity errors from Scenario 3. Note that only the simulations with a 3DRMS below 10 m are shown.

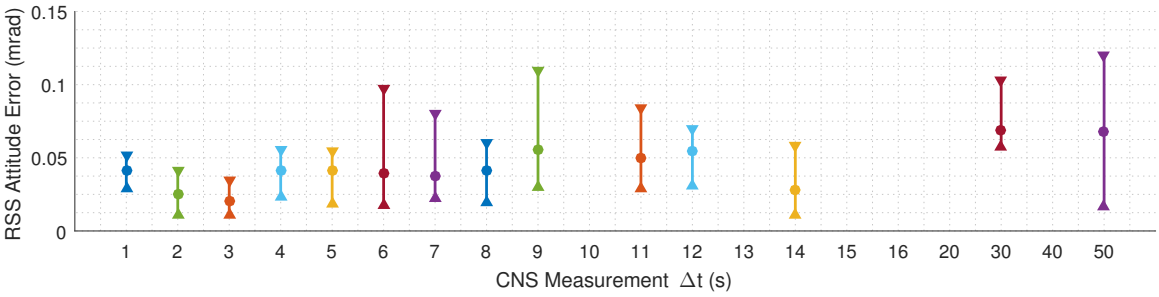


Figure 36. CDF error bars of the RSS attitude errors from Scenario 3. Note that only the simulations with a 3DRMS below 10 m are shown.

4.4 Comparisons of Results

The previous sections analyzed the results within each scenario to understand how observation times affected the navigation estimates, which for the most part were uncorrelated. This section compares the results between scenarios to show what kind of improvements are seen using different sensor configurations. Most of the comparisons will be between Scenarios 2 and 3 due to the large drift errors seen in Scenario 1, which would make the plots difficult to read if included in the comparison. The results of Scenario 1 can still be compared with the other scenarios in Table 4.

Figure 37 overlays the 3DRMS accuracies from Scenario 2 and Scenario 3. The position accuracy between the two are nearly identical except for the two outlying results from Scenario 2. It's clear that the position estimates are nearly identical between the two scenarios.

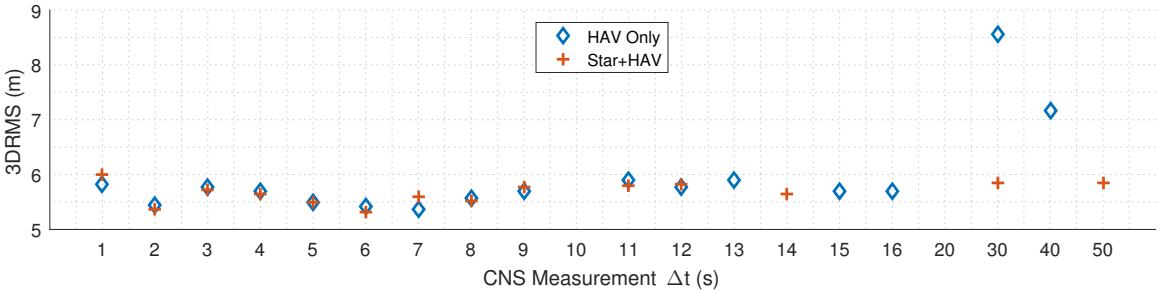


Figure 37. Comparison of 3DRMS accuracy between Scenario 2 and Scenario 3.

Another measure of overall accuracy is by taking the RSS of the standard deviation

of the errors in each component. Figure 38 shows the RSS of error σ for position, velocity, and attitude between Scenario 2 and 3. The position and velocity accuracies are nearly the same with the exception of the outlying runs for Scenario 2, and would be identical without those two outliers in Scenario 2. The only noticeable difference between the two scenarios is in the attitude, in which Scenario 3 is about 8 times more accurate than Scenario 2.

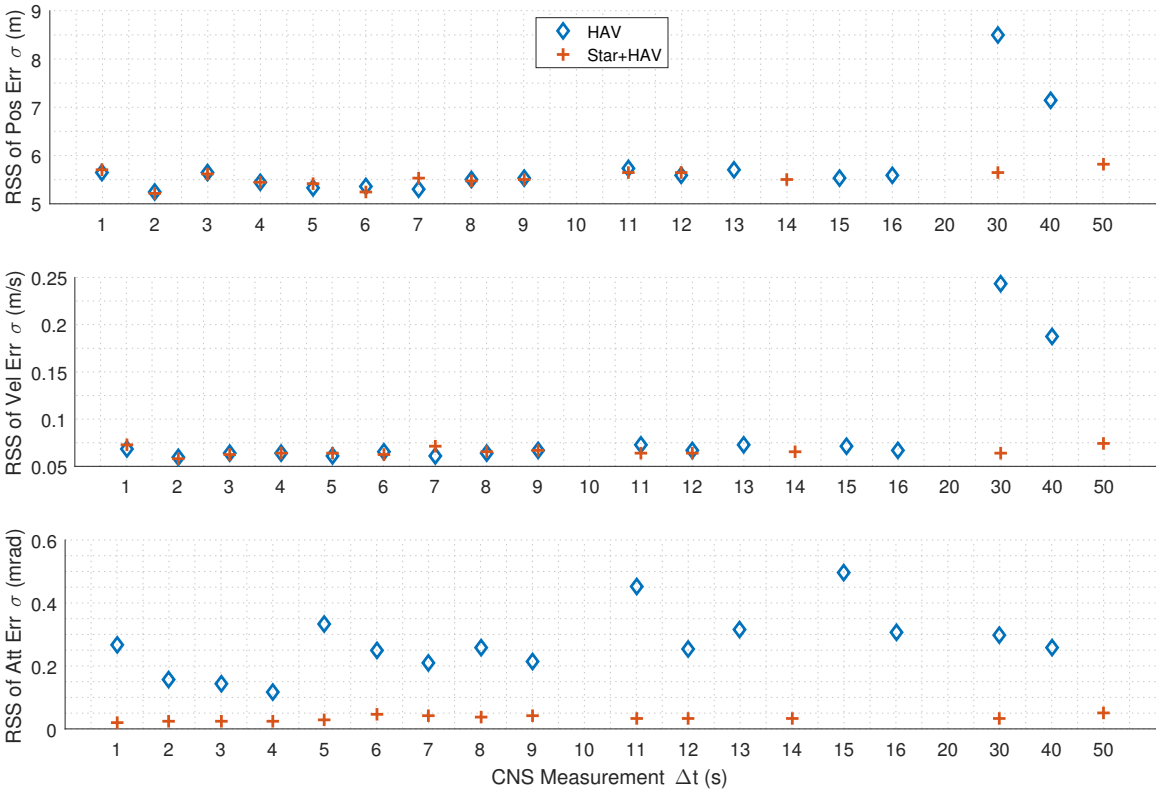


Figure 38. Comparison of the RSS of the navigation error standard deviations between Scenario 2 and Scenario 3.

Given the conclusion that CNS measurement frequency doesn't affect the results much at all, the results of all 20 simulations per scenario is no different than running 20 simulations of the same scenario. As such, the mean of the results can be compared for a singular value to rate the entire scenario. Table 4 summarizes the results of Figures (37) and (38) by taking the means of the results. In addition to the results of the scenarios, a control scenario was run to show the results of the simulation

without any CNS, in which the altimeter was the only sensor used to aid the INS. Because there were no parameters to change in the control, only 10 simulation runs were generated instead of 20 to determine the mean and standard deviation values between runs.

Table 4. Comparison of 3DRMS and RSS of error σ in position, velocity, and attitude between all three scenarios. Row C represents the control simulation in which the INS was allowed to drift with only the altimeter available for updates.

	3DRMS		**RSS of Error σ**		
	Mean (m)	**σ (m)**	**Position (m)**	**Velocity (m/s)**	**Attitude (mrad)**
C	10,751.41	4722.42	4176.23	6.31	0.77
S1	7715.34	3556.15	3286.06	4.73	0.57
S2	5.93	0.81	5.80	0.085	0.27
S3	5.67	0.19	5.53	0.066	0.034

The star tracker shows a slight improvement in 3DRMS by 28% compared to no CNS, as well as small improvements in velocity and attitude. However it's clear that by itself the star tracker is not a reliable method of navigation as the position errors are still in the thousands of meters. Introducing HAV tracker provides more than 99.9% improvement in 3DRMS position accuracy. Surprisingly, the HAV tracker also shows better attitude accuracy than the star tracking, with a 50% improvement in attitude. Scenarios 2 and 3 have comparable results, but the main performance benefit of the combined star and HAV tracker configuration is the significant attitude accuracy it provides over all the other sensor configuration, with an 87% improvement.

4.5 Chapter IV Summary

The most surprising result from these simulations is that the time between measurements had no significant impact to the navigation accuracy in all three scenarios. Based on the results from all three scenarios, there is little navigational performance benefits, if any, from having very frequent stellar or HAV measurements. This is

most likely due to the fact that a navigation grade IMU was used, limiting the drift between measurements even up to 50 s. As for the configuration comparison, it was expected that star tracking only provided poor position estimates compared to HAV tracking but it was unexpected that HAV tracking provided better attitude certainty than star tracking. Introducing the HAV tracker provides a tremendous boost in position and velocity estimate accuracy. The last configuration of using both sensors provides minimal position and velocity improvements, but attitude accuracy is an order of magnitude greater than either sensor individually.

V. Conclusion

This chapter concludes this paper with a summary of the work presented, highlighting the contributions of the results in the field celestial navigation as well as some difficulties that were encountered. Additionally, some potential future work is discussed, growing from unanswered questions from this research and other potential avenues for improvement.

5.1 Summary of the Document

Chapter I provided an introduction as to why celestial navigation is relevant in the field of alternative navigation solutions. A brief overview of how CNS works was given, with the traditional use of star sensing for attitude determination as well as position and velocity determination from observations of a known reference object. It provided relevant research in similar areas of celestial navigation, describing the works and results of others in both space-born applications of CNS and CNS within the Earth's atmosphere. Chapter I concluded with the organization of the remainder of the paper and an introduction to the MATLAB simulation tool, CAINS.

Chapter II provided a deeper understanding of celestial navigation. It described how the star tracker determines the star's PSF center through a process known as centroiding with sub-pixel level accuracy, and how that affects the overall attitude accuracy. A description of CNS absolute triangulation was provided, allowing the determination of position and velocity by observing known reference and background objects. Assuming that the foreground object needs to be a point source, the optics theory for getting the reference to appear as a point source was discussed. Chapter II concluded with a general description of EKF propagation and update equations.

Chapter III described the research methodology. It described the different sce-

narios that were simulated to show the effects of observation measurement frequency and multiple sensor configurations on total navigation accuracy. Key assumptions were laid out and discussed. The process flow for CAINS showed how the sensor measurements are incorporated into the EKF state estimates, and the specific linearized models were shown. Finally, the generation of the measurement data files was discussed.

The results of the simulations were shown in Chapter IV. From the analysis, it turns out that there is no significant performance improvement by increasing the observation frequencies from 1 s to 50 s in all three scenarios. The performance in position and velocity estimation improves dramatically with the addition of the HAV tracker, providing over 99.9% position and velocity estimates. If accurate attitude is required though, the combined star and HAV tracker configuration is necessary as that is the only way to reach arcsecond level attitude accuracies. The results also showed some confusing behavior from the EKFs anomaly detection algorithm, though it may be an issue with the actual simulation setup rather than the filter. Fortunately the number of anomalous simulations were few enough to allow them to be discarded and still generate good data from the remainder.

5.2 Future Work

This research may be continued in a number of avenues. The first is the removal of the simplifications due to the first two assumptions. Turbulence is a non-negligible factor that will decrease the navigation aiding estimates. Additionally, there will be a time delay between the visual HAV measurement and receiving communication of the HAV position. This will introduce error based on the length of the time lag. This can be mitigated in two ways: by estimating the time lag through the EKF, which will require adding additional states to the filter, or by delaying the update algorithm

until the HAV's location is received and propagate the solution to the current time after the fact.

Another area of future research would be to simulate a gimbaled CNS rather than a fixed CNS. This will allow this simulation to overcome certain limitations of the current setup. The first is that it will allow the HAV to have more realistic flight trajectories rather than being forced to fly directly above the RPY to stay within its FOV. This allows multiple RPAs to view one HAV, or one RPA to get measurements from multiple HAVs. It can also overcome the fact that certain attitude fluctuations will impede taking measurements of the HAV. This will allow more rigorous work to be performed on measurement frequency to determine if Δt's greater than 50 s will affect navigation aiding.

The issue with the anomaly detection algorithm is left unanswered in this research. Further investigation should be done to explain why the EKF flags certain measurements as anomalous, even though the input measurements were identical. While it didn't have much of an impact in the results of this research, it could be an issue if left unresolved for future work.

Bibliography

1. Jamshaid Ali and Jiancheng Fang. Realization of an autonomous integrated suite of strapdown astro-inertial navigation systems using unscented particle filtering. *Computers and Mathematics with Applications*, 57(2):169–183, 2009.

2. Humood Alkhaldi. Integration of a Star Tracker and Inertial Sensors Using an Attitude Update. Master's thesis, Air Force Institute of Technology, 2014.

3. Jorge E. Diaz. Satellite Ephemeris Correction via Remote Site Observation for Star Tracker Navigation Performance Improvement. Master's thesis, Air Force Institute of Technology, 2016.

4. Tom Dzamba and John Enright. Optical trades for evolving a small arcsecond star tracker. *IEEE Aerospace Conference Proceedings*, 2013.

5. Joseph W. Goodman. *Introduction to Fourier Optics*. Roberts & Company, Colorado, 3 edition, 2005.

6. George H. Kaplan. Angles-Only Navigation: Position and Velocity Solution from Absolute Triangulation. *Journal of The Institute of Navigation*, 58(3):187–201, 2011.

7. John D. Kraus. *Radio Astronomy*. McGraw-Hill Book Company, New York, 1966.

8. Carl Christian Liebe. Accuracy Performance of Star Trackers - A Tutorial. *IEEE Transactions on Aerospace and Electronic Systems*, 38(2):578–599, 2002.

9. Peter S. Maybeck. *Stochastic Models, Estimation, and Control: Volume 1*. Academic Press, New York, 1 edition, 1979.

10. Peter S. Maybeck. *Stochastic Models, Estimation, and Control: Volume 2*. Academic Press, New York, 1 edition, 1982.

11. Pratap Misra and Per Enge. *Global Positioning System: Signals, Measurements, and Performance*. Ganga-Jamuna Press, Massachusetts, 2 edition, 2012.

12. Xiaolin Ning and Jiancheng Fang. An autonomous celestial navigation method for LEO satellite based on unscented Kalman filter and information fusion. *Aerospace Science and Technology*, 11(2-3):222–228, mar 2007.

13. H. Nobahari, H. Ghanbarpour Asl, and S. F. Abtahi. A back-propagation approach to compensate velocity and position errors in an integrated inertial/celestial navigation system using unscented Kalman filter. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 228(10):1702–1712, 2014.

14. F. Pappalardi, S.J. Dunham, M.E. LeBlang, T.E. Jones, J. Bangert, and G. Kaplan. Alternatives to GPS. *MTS/IEEE Oceans 2001. An Ocean Odyssey. Conference Proceedings*, 3:1452–1459, 2001.

15. Scott J. Pierce. *Modeling Navigation System Performance of a Satellite-Orbiting Star Tracker Tightly Integrated with an Inertial Measurement Unit*. PhD thesis, Air Force Institute of Technology, 2015.

16. Amir Moghtadaei Rad, Jafar Heyrani Nobari, and Amir Ali Nikkhah. Optimal Attitude and Position Determination by Integration of INS, Star Tracker, and Horizon Sensor. *IEEE Aerospace and Electronic Systems Magazine*, 29(4):20–33, 2014.

17. Richard D. Richmond and Stephen C. Cain. *Direct-Detection LADAR Systems*. SPIE, Washington, 1 edition, 2010.

18. Sean E. Urban and Kenneth P. Seidelmann. *Explanatory Supplement to the Astronomical Almanac*. University Science Books, California, 3 edition, 2013.

19. David A. Vallado. *Fundamentals of Astrodynamics and Applications*. Microcosm Press and Kluwer Academic Publishers, California and Dordrecht, 2 edition, 2004.

20. J. Chris Zingarelli, Eric Pearce, Richard Lambour, Travis Blake, Curtis J. R. Peterson, and Stephen Cain. Improving the Space Surveillance Telescope'S Performance Using Multi-Hypothesis Testing. *The Astronomical Journal*, 147(5), 2014.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
23–03–2017	Master's Thesis	Sept 2015 — Mar 2017

4. TITLE AND SUBTITLE

Celestial Aided Inertial Navigation by Tracking High Altitude Vehicles

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Kim, Mark, S., Capt, USAF

5d. PROJECT NUMBER

17G743

5e. TASK NUMBER

5f. WORK UNIT NUMBER

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

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

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-MS-17-M-040

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

Air Force Research Laboratory, Munition's Directorate
Attn: Kevin M. Brink
101 West Eglin Blvd
Eglin AFB, FL 32542
850-872-4600
Email: kevin.brink@us.af.mil

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/RW

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

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A:
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

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

14. ABSTRACT

Celestial and inertial navigation systems work well together as an alternative to Global Positioning System. Inertial systems provide constant dead reckoning updates but is subject to drift. Celestial systems provide updates with its passive stellar measurements to correct the inertial drift. Stellar measurements normally update attitude only by tracking the angular positions of known stars. However, by tracking reference objects with known positions against a background of stars, the observers position and velocity can be updated as well. Using a MATLAB tool developed by the Air Force Research Laboratory, this research simulates the navigation performance of a low flying aircraft tracking a higher flying aircraft as the reference object. Three different scenarios are studied: 1) stellar observations providing attitude updates only, 2) aircraft observations providing bearing measurements to known position and velocities, and 3) both stellar and aircraft observations. Additionally, the observation frequency will be a variable parameter to determine its effect on navigation accuracies. The sensor measurements are combined using an extended Kalman filter.

15. SUBJECT TERMS

Celestial Navigation, Alternative Navigation

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Stephen C. Cain, AFIT/ENG
U	U	U	UU	81	19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636, x4716; Stephen.Cain@afit.edu